Πανεπιστημιο Πατρων
Τμημα Μηχανικων Η/Υ και Πληροφορικης

Διπλωματικη Εργασια

Αλγοριθμοι και Τεχνικες Δημιουργιας Περιληψης Κειμενου και Εφαρμογη τους σε Συσκευες Μικρου Μεγεθους

Τσόγκας Βασίλειος
Α.Μ. 2974

Υπεύθυνος Καθηγητής:
Χρήστος Μπούρας,
Αναπληρωτής Καθηγητής

Αυγουστος 2007

'Αφιερωμένη στους ανθρώπους που με αγαπούν,
με στηρίζουν, με ανέχονται και είναι πάντα δίπλα μου,
στην οικογένειά μου'

Πρόλογος

Ζούμε σε μια κοινωνία αλλαγής και προόδου. Σε μια κοινωνία που χαρακτηρίζεται από τον τεράστιο όγκο της πληροφορίας που διακινείται μέσα στις τάξεις της. Κυρίως όμως διανύουμε την εποχή της κατάργησης των συνόρων και της αδιάλειπτης επικοινωνίας μεταξύ των ανθρώπων. Το διαδίκτυο αποτελεί τον τροχό γι' αυτές τις αλλαγές· η ποσότητα όμως των δεδομένων που υπάρχουν και διακινούνται μέσω αυτού είναι τόσο τεράστια, ώστε να αποσπά τους πολίτες της κοινωνίας αυτής στην προσπάθειά τους να βρουν χρήσιμη πληροφορία και επομένως να μετατρέπεται σε τροχοπέδη της αλλαγής. Παράλληλα, η ολοένα και επεκτεινόμενη χρήση συσκευών μικρού μεγέθους για πλοήγηση στον παγκόσμιο ιστό, κάνει περισσότερο εμφανή τα μεγάλα προβλήματα κατά την προσπάθεια αναζήτησης, πρόσβασης και ανάγνωσης της πληροφορίας.

Με την πραγματικότητα των υπέρογκων και ολοένα αυξανόμενων πηγών κειμένου στο διαδίκτυο, καθίστανται αναγκαία η ύπαρξη μηχανισμών οι οποίοι βοηθούν τους χρήστες ώστε να λάβουν γρήγορες απαντήσεις στα ερωτήματά τους. Η παρουσίαση προσωποποιημένου, συνοψισμένου και προκατηγοριοποιημένου περιεχομένου στους χρήστες, κρίνεται απαραίτητη σύμφωνα με τις επιταγές της συνδυαστικής έκρηξης της πληροφορίας που είναι ορατή σε κάθε 'γωνία' του διαδικτύου. Ζητούνται άμεσες και αποτελεσματικές λύσεις ώστε να 'τιθασευτεί' αυτό το χάος πληροφορίας που υπάρχει στον παγκόσμιο ιστό, λύσεις που είναι εφικτές μόνο μέσα από ανάλυση των προβλημάτων και εφαρμογή σύγχρονων μαθηματικών και υπολογιστικών μεθόδων για την αντιμετώπισή τους.

Επιτελική σύνοψη

Η εξάπλωση του Διαδικτύου είναι τεράστια και η πληροφορία που διακινείται είναι αχανής. Αυτό έχει ως αποτέλεσμα η ανεύρεση της πληροφορίας από τους χρήστες να είναι μία διαδικασία εξαιρετικά χρονοβόρα και επίπονη. Ο μέσος χρήστης ενός υπολογιστή σπαταλά ελάχιστο χρόνο για να αναγνώσει κείμενα τα οποία ξεπερνούν τη μία σελίδα και συχνά απορρίπτει πληροφορία η οποία ενδέχεται να είναι πολύ χρήσιμη σε αυτόν.

Στα πλαίσια της συγκεκριμένης διπλωματικής εργασίας, δημιουργήθηκε ένας ολοκληρωμένος μηχανισμός ο οποίος μπορεί αυτόματα να κάνει λεξικογραφική ανάλυση ενός κειμένου προκειμένου να εξάγει λέξεις-κλειδιά. Μέσα από αυτή την ανάλυση προκύπτουν οι προτάσεις του κειμένου που το χαρακτηρίζουν και οι οποίες μπορούν, αν συνενωθούν, να αποτελέσουν μια σύντομη περίληψη του κειμένου. Μέσα από τεχνικές και αλγορίθμους δημιουργίας περίληψης κειμένου καθίσταται εφικτή η δημιουργία διαφορετικού μήκους περιλήψεων, κάθε μία από τις οποίες μπορεί να αποστέλλεται σε συσκευές διαφορετικού μεγέθους, ανταποκρινόμενοι στο κάλεσμα της εποχής για σύνοψη και αξιοποίηση της χρήσιμης μόνο πληροφορίας μέσα στη χαοτική κατάσταση που επικρατεί στο διαδίκτυο.

Ο μηχανισμός που αναπτύχθηκε δεν αποτελεί ένα μονοδιάστατο, στάσιμο ή 'κλειστό' σύστημα. Αντίθετα, έγινε κάθε προσπάθεια ώστε να είναι επεκτάσιμος, δυναμικός και 'ανοιχτός' σε νέα δεδομένα και προκλήσεις. Είναι τμηματοποιημένος, επιτυγχάνοντας πολλές διεργασίες ανάκτησης και εξόρυξης πληροφορίας ανεξάρτητα μεταξύ τους, έχοντας τη δυνατότητα συνδυασμού τους για την επίτευξη καλύτερων αποτελεσμάτων. Κατασκευάστηκε εξ' αρχής μηχανισμός εξαγωγής κωδικολέξεων, στάδιο βασικό για κάθε τεχνική ανάκτησης πληροφορίας. Αναπτύχθηκε μηχανισμός κατηγοριοποίησης κειμένου, ο οποίος μπορεί να συμμετάσχει στη διαδικασία εξαγωγής περίληψης και να ενδυναμώσει τα αποτελέσματά της. Εξελίχθηκαν αλγόριθμοι και μηχανισμοί προσωποποίησης περιεχομένου στο χρήστη για την περαιτέρω βελτίωση της εξαγόμενης περίληψης. Μελετήθηκαν θέματα που έχουν να κάνουν με χρήστες συσκευών μικρού μεγέθους και αναπτύχθηκε υποσύστημα εξαγωγής περίληψης προσαρμοσμένης στα μέτρα αυτών των συσκευών. Πάνω από όλα όμως, η συγκεκριμένη διπλωματική εργασία είχε έντονο ερευνητικό χαρακτήρα, μελετώντας τις υπάρχουσες εξελίξεις στα θέματα με τα οποία καταπιάνεται και αξιοποιώντας προκλήσεις και ιδέες για την πρόταση και υλοποίηση ενός καινοτόμου συστήματος.

Η ερευνητική διατριβή που έγινε στα πλαίσια της διπλωματικής εργασίας οδήγησε στις παρακάτω δημοσιεύσεις:

Διεθνή περιοδικά

- PeRSSonal's core functionality evaluation: enchancing text labeling through personalized summaries. Elsevier, Data & knowledge engineering (DKE) journal, C. Bouras, V. Poulopoulos, V. Tsogkas (to appear)
  abstract: Σε αυτή τη δημοσίευση παρουσιάζουμε τα υποσυστήματα περίληψης και κατηγοριοποίησης ενός αλοκληρωμένου μηχανισμού που ξεκινά από το κατέβασμα ιστοσελίδων και ολοκληρώνεται με την παρουσίαση των συγκεντρωμένων δεδομένων στους τελικούς χρήστες μέσω ενός προσωποποιημένου portal. Το σύστημα στοχεύει στη συλλογή άρθρων από ιστοσελίδες γνωστών ειδησεωγραφικών πρακτορείων του παγκοσμίου ιστού και, ακολουθόντας μια αλγοριθμική διαδικασία, να δημιουργήσει μια πιο φιλική και προσωποποιημένη 'άποψη' των άρθρων. Πριν παρουσιαστούν οι πληροφορίες πίσω στον χρήστη, ο πυρήνας του μηχανισμού κατηγοριοποιεί αυτόματα τα κείμενα και στη συνέχεια

εξάγει τις περιλήψεις τους. Επικεντρωνόμαστε στον πυρήνα του μηχανισμού και πιο συγκεκριμένα, παρουσιάζουμε τους αλγορίθμους που χρησιμοποιούνται για την περίληψη και την κατηγοριοποίηση των κειμένων. Οι αλγόριθμοι δεν αξιοποιούνται μόνο για την παραγωγή απομωνομένων δεδομένων, στοχευμένα για ένα συγκεκριμένο υποσύστημα, αλλά ένας συνδιασμός αλγορίθμων, οι οποίοι επιτυγχάνουν τη συνεργασία των υποσυστημάτων κατηγοριοποίησης και περίληψης, εισάγεται με σκοπό να βελτιωθεί η κατηγοριοποίηση κειμένων μέσω των προσωποποιημένων εξαγμένων περιλήψεων.

Διεθνή συνέδρια

- Efficient Summarization Based On Categorized Keywords. The 2007 International Conference on Data Mining (DMIN'07), Las Vegas, Nevada, USA, C. Bouras, V. Poulopoulos, V. Tsogkas, 25 - 28 June 2007
  abstract: Η πληροφορία που υπάρχει στον παγκόσμιο ιστό είναι τόσο τεράστια ώστε να αποσπά τους χρήστες από την προσπάθειά τους να βρουν χρήσιμη πληροφορία. Για να ξεπεραστεί το πρόβλημα, πολλοί μηχανισμοί προσωποποίησης και περίληψης έχουν παρουσιαστεί. Σε αυτή τη δημοσίευση προτείνουμε έναν μηχανισμό ο οποίος εφαρμόζει τεχνικές περίληψης άρθρων τα οποία εξάγονται από τον παγκόσμιο ιστό, βασισμένος στην διαδικασία κατηγοριοποίησης (η οποία εφαρμόζεται επίσης στα ίδια άρθρα). Μέσα από εκτεταμένο πειραματισμό, αποδεικνύουμε ότι η διαδικασία περίληψης κειμένου επιρρεάζει το μηχανισμό κατηγοριοποίησης και αντιστρόφως. Αυτό σημαίνει πως, όταν τα αποτελέσματα του μηχανισμού περίληψης είναι ασθενή, τόστε η κατηγοριοποίηση κειμένου μπορεί να χρησιμοποιηθεί για την βελτίωση της εξαγόμενης περίληψης και, από την άλλη μεριά, όταν η διαδικασία κατηγοριοποίησης υπερφορτώνεται λόγω μεγέθους, τα περιληπτημένα άρθρα μπορούν να χρησιμοποιηθούν για την πιο αποτελεσματική κατηγοριοποίηση των άρθρων. Παράλληλα, σε αυτή τη δημοσίευση εκφράζεται ότι ο συνδιασμός της περίληψης και κατηγοριοποίησης μπορεί να οδηγήσει σε καλύτερα αποτελέσματα, όχι μόνο και για τους δύο μηχανισμούς, αλλά και για το προσωποποιημένο portal. Τέλος, προτείνουμε έναν ολοκληρωμένο μηχανισμό ο οποίος μπορεί να χρησιμοποιηθεί για να παρέχει στους χρήστες τα απαραίτητα εργαλεία με σκοπό τον ευκολότερο εντοπισμό της πληροφορίας που χρειάζονται.

- Personalizing text summarization based on sentence weighting. IADIS European First International Conference Data Mining (ECDM 2007), Lisbon, Portugal, C. Bouras, V. Poulopoulos, V. Tsogkas, 3 - 8 July 2007
  abstract: Η ποσότητα των δεδομένων του υπάρχουν στο διαδίκτυο είναι τόσο υπέρογκη ώστε να εμποδίζει τους χρήστες στην προσπάθειά τους για αναζήτηση χρήσιμης πληροφορίας. Παράλληλα, η ολοένα και εκτεινόμενη χρήση συσκευών μικρού μεγέθους για πλοήγηση στον παγκόσμιο ιστό δημιουργεί τεράστια προβλήματα κατά την προσπάθεια αναζήτησης και ανάγνωσης πληροφορίας. Μια λύση γι΄ αυτά τα θέματα είναι η προσωποποίηση του ιστού και η προσπάθεια για αλγοριθμική μείωση της ποσότητας κειμένου. Πολλοί περιλήπτες κειμένου έχουν παρουσιαστεί σε μια προσπάθεια μείωσης της άχρηστης πληροφορίας που παρουσιάζεται στους χρήστες και πολλές ιστοσελίδες, ειδικά τα news portals, εισάγουν χαρακτηριστικά προσωποποίησης για τους χρήστες, αν και ακόμη αυτές οι τεχνικές δεν χρησιμοποιούνται σε συνδιασμό με άλλες για την παραγωγή ακόμα καλύτερων αποτελεσμάτων. Σε αυτή τη δημοσίευση παρουσιάζεται ένας μηχανισμός δημιουργίας προσωποποιημένων περιλήψεων για τα μέλη ενός news portal ο οποίος αναπαράγει άρθρα τα οποία συλλέγονται από μεγάλα news portals του διαδικτύου, καθώς και την αξιολόγηση τόσο του περιλήπτη όσο και των προσωποποιημένων περιλήψεων. Οι αξιολογητές των προσωποποιημένων περιλήψεων είναι μέλη του news portal. Ο μηχανισμός προσωποποιημένης περίληψης μπορεί επίσης να χρησιμοποιηθεί από χρήστες συσκευών μικρού μεγέθους, για ανετότερη ανάγνωση λιγότερης αλλά πιο περιεκτικής πληροφορίας.

- The importance of the difference in text types to keyword extraction: Evaluating a mechanism. 7th International Conference on Internet Computing 2006 (ICOMP 2006), Las Vegas, Nevada, USA, C. Bouras, C. Dimitriou, V. Poulopoulos, V. Tsogkas, 26 - 29 June 2006
  abstract: Η πληροφορία υπάρχει και κάθε άποψη της ζωής μας. Η επέκταση του παγκοσμίου ιστού έχει βοηθήσει προς αυτή την κατεύθυνση. Ο παγκόσμιος ιστός μας 'ταϊζει' με τεράστια ποσότητα πληροφορίας και η εκτεταμένη χρήση των υπολογιστών και άλλων συσκευών μας έχει οδηγήσει σε μια κατάσταση όπου παρότι έχουμε πολύ διαθέσιμη πληροφορία στα χέρια μας, τις περισσότερες φορές μας είναι άχρηστη. Οι άνθρωποι δυσκολεύονται να εντοπίσουν πληροφορία που χρειάζονται και στην ουσία κατέχουν. Πόσες φορές δεν έχουμε προσπαθήσει να εντοπίσουμε ένα συγκεκριμένο

άρθρο ή ένα συγκεκριμένο μήνυμα που λάβαμε ή κάποιο SMS που γνωρίζουμε ότι κατέχουμε. Γι' αυτούς τους λόγους πολλές τεχνικές ανάκτησης πληροφορίας έχουν προταθεί και πολλοί μηχανισμοί εξαγωγής πληροφορίας έχουν δημιουργηθεί. Σε αυτή τη δημοσίευση παρέχουμε την πειραματική αξιολόγηση ενός μηχανισμού εξαγωγής κωδικολέξεων και παρουσιάζουμε την διαφορετική αντιμετώπιση που έχουμε για τα διάφορα είδη κειμένων (άρθρα νέων, δημοσιεύσεις και e-mails). Αυτός ο μηχανισμός εξαγωγής κωδικολέξεων είναι μέρος ενός πλήρους συστήματος που περιλαμβάνει υποσυστήματα ανάκτησης πληροφορίας, εξαγωγή πληροφορίας, κατηγοριοποίησης και δημοσίευση πληροφορίας σε προσωποποιημένο portal.

Κλείνοντας, θα ήθελα να ευχαριστήσω θερμά τον καθηγητή μου κ. Χρήστο Μπούρα (Αναπληρωτής Καθηγητής του τμήματος Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής του Πανεπιστημίου Πατρών) για την επίβλεψη της εργασίας, την πολύτιμη βοήθειά του και τη συμπαράστασή του προκειμένου να ολοκληρωθεί με επιτυχία η διπλωματική εργασία αλλά κυρίως για την ευκαιρία που μου έδωσε να ασχοληθώ με τα τόσο ενδιαφέροντα ερευνητικά πεδία με τα οποία καταπιάνεται η εργασία.

Θα ήθελα επίσης να ευχαριστήσω τον καλό μου φίλο Βασίλη Πουλόπουλο για την συμπαράσταση, την καθοδήγηση, την βοήθειά του, αλλά και τη συνεργασία που είχαμε καθ' όλη τη διάρκεια εκπόνησης της διπλωματικής μου εργασίας.

Περιεχόμενα

Κατάλογος Πινάκων

Κατάλογος Σχημάτων

Γλωσσάρι

Association pattern	Πρότυπο συσχέτισης
Boolean	Δυαδική λογική
Browser	Φυλλομετρητής Ιστού
Categorization	Κατηγοριοποίηση
Classification	Ταξινόμηση
Content	Περιεχόμενο
Corpus	Συλλογή κειμένων με περιλήψεις αυτών
Crawler, bot, spider	Μηχανισμοί που πραγματοποιούν διαπέρασμα των σελίδων του Διαδίκτυου
Data mining	Εξόρυξη δεδομένων
Decision tree	Δέντρο απόφασης
E-mail	Ηλεκτρονικό Ταχυδρομείο
Efficient	Αποτελεσματικός
Embedded software	Ενσωματωμένο λογισμικό
Flexible	Ευέλικτος
Format	Μορφοποίηση
Front-end	Περιβάλλον αλληλεπίδρασης με το χρήστη
Fuzzy	Ασαφές
Generic	Γενικού περιεχομένου
HTML Editor	Πρόγραμμα με το οποίο μπορεί να γίνει επεξεργασία HTML γλώσσας
Html	Η βασική γλώσσα του διαδίκτυου
Information Filtering	Φιλτράρισμα Πληροφορίας
Information Retrieval	Ανάκτηση Πληροφορίας
Internet	Διαδίκτυο
Keywords	Κωδικολέξεις
Knowledge mining	Εξόρυξη γνώσης
Link	Σύνδεσμος (αναφέρεται σε ιστοσελίδα)
Machine Understandable	Κατανοητός από μηχανή
Metadata	Μεταδεδομένα
Module	Τμήμα, κομμάτι
NLP	Επεξεργασία φυσικής γλώσσας
News portals	Ειδησεογραφικες ιστοσελίδες
Ontology	Οντολογία, αντικείμενο
Portable	Φορητός
Portal / News Portal	Δικτυακή πύλη / Δικτυακή πύλη ενημερωτικού περιεχομένου
Preprocessing	Προεπεξεργασία
Punctuation	Στίξη

RSS feed	Τροφοδοσία περιεχομένου με το πρότυπο RSS
Search engine persuasion	Πειθώ των μηχανών αναζήτησης
Semantic Web	Σημασιολογικός ιστός
State of the Art	Οι τρέχουσες εξελίξεις στην επιστήμη
Stemmer	Πρόγραμμα που εφαρμόζει τη διαδικασία εξαγωγής ρίζας μιας λέξης
Stemming	Διαδικασία εύρεσης της ρίζας μίας λέξης
Stopwords	Αναφέρεται σε συγκεκριμένη λίστα λέξεων η οποία περιέχει λέξεις που πρέπει να διαγραφούν από τις λέξεις κλειδιά ενός κειμένου γιατί θεωρούνται κοινότυπες.
Tag	Επικεφαλίδα. Ο όρος χρησιμοποιείται για τις δηλώσεις σε HTML γλώσσα
Text Analysis	Ανάλυση κειμένου
Text Categorization	Κατηγοριοποίηση κειμένου
Training set	Σύνολο εκμάθησης
User profile	Προφίλ χρήστη
Vector Space Model	Μοντέλο κατηγοριοποίησης που στηρίζεται στη χρήση διανυσμάτων και πινάκων
WWW	Παγκόσμιος Ιστός

Συντομογραφίες

DBMS	DataBase Management System
HTML	HyperText Mark-up Language
IF	Information Filtering
IR	Information Retrieval
LSI	Latent Semantic Indexing
RSS	Rich Site Summary
SVM	Support Vector Machine
URL	Uniform Resource Locator
VSM	Vector Space Model
WWW	World Wide Web
ΑΠ	Ανάκτηση Πληροφορίας
ΒΔ	Βάση Δεδομένων
ΠΣ	Πληροφοριακό Σύστημα

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

> Dignity consists not in possessing honors, but in the consciousness that we deserve them.
>
> Aristotle, Greek critic, philosopher, physicist, & zoologist

Η εξάπλωση του παγκοσμίου ιστού είναι τεράστια και καθημερινά όλο και περισσότεροι άνθρωποι στον κόσμο γίνονται μέλη της κοινότητας του Διαδικτύου. Οι τεχνολογίες άλλωστε που αφορούν το Διαδίκτυο γνωρίζουν τρομερή άνθιση τα τελευταία χρόνια και πολλές ερευνητικές δραστηριότητες τείνουν να προσεγγίσουν από κάθε πλευρά, τεχνολογική, στατιστική, καθαρά μαθηματική ή και κοινωνική, την ολοένα αυξανόμενη και συνάμα παράξενη κοινότητα.

Το Internet είναι πλέον παντού. Έχει επιτύχει την παγκοσμιότητα που άλλωστε επιτάσσει το όνομά του: World Wide Web. Σχεδόν κάθε συσκευή, σταθερή ή κινητή μπορεί να έχει τη δυνατότητα να συνδεθεί στο Διαδίκτυο. Συνεπώς το πρόβλημα της εύκολης πρόσβασης στον παγκόσμιο ιστό θεωρείται ίσως ξεπερασμένο, ή καλύτερα δεν δίνεται πλέον τόσο μεγάλο βάρος σε αυτό. Αντίθετα, το βάρος δίνεται πλέον στην κοινότητα του Διαδικτύου. Στην επονομαζόμενη και 'κοινωνία της πληροφορίας'. Στην 'κοινωνία' αυτή του Διαδικτύου, οι χρήστες είναι πλέον ενεργά μέλη. Το αρχικό 'μοντέλο' της εύρεσης κάθε είδους πληροφορίας στον παγκόσμιο ιστό έχει ξεπεραστεί προ πολλού. Τα ενεργά μέλη πλέον επιθυμούν όχι μόνο να εξορύξουν πληροφορία αλλά και να δημιουργήσουν το δικό τους προσωπικό χώρο όπου κανείς θα μπορεί να βρει επιπρόσθετη της ήδη υπάρχουσας πληροφορίας. Η κατάσταση είναι πλέον χαοτική αν όχι δραματική. Το περιεχόμενο που διακινείται στον παγκόσμιο χώρο του Διαδικτύου είναι απέραντο, σε βαθμό που να μην είναι εφικτός ο προσδιορισμός του μεγέθους του. Το γεγονός αυτό μπορεί να περιέχει πολλά θετικά στοιχεία, συνάμα όμως δημιουργεί βασικές δυσάρεστες παρενέργειες.

1.1 Προσδιορισμός του προβλήματος

Όπως κάθε κοινωνία, έτσι και το Διαδίκτυο, έχει τα δικά του προβλήματα. Πηγή αυτών των προβλημάτων μπορεί να θεωρηθεί η 'άναρχη δόμησή του', η έλλειψη σαφούς νομοθεσίας αλλά και η αίσθηση ελευθερίας που αφήνει τους 'κατοίκους' του να ενεργούν ουσιαστικά κατά βούληση, βρίσκοντας στο Διαδίκτυο μία επανάσταση που θέλουν στην πραγματική τους ζωή, έναν τρόπο έκφρασης ιδεών, έναν τρόπο έκφρασης της γνώσης και της μάθησης.

Η ελευθερία της έκφρασης και του λόγου παγκοσμίως διασφαλίζεται πλέον από τον τρόπο με τον οποίο διακινείται περιεχόμενο στο Διαδίκτυο. Η διάχυση γνώσης και εμπειρίας θα μπορούσαν επίσης να χαρακτηριστούν σαν θετικά επακόλουθα από την ύπαρξη μεγάλου όγκου πληροφορίας στον παγκόσμιο ιστό. Θα πρέπει όμως κανείς να αναλογιστεί κατά πόσο όλος αυτός ο όγκος πληροφορίας και όλες οι πηγές ενημέρωσης του Διαδικτύου είναι έγκυρες. Δεν υπάρχει απολύτως κανένας μηχανισμός που να μπορεί να

διασφαλίσει σε κάθε επισκέπτη του Διαδικτύου πως οι σελίδες που παρακολουθεί και το περιεχόμενο που συλλέγει είναι αξιόπιστο και ποιοτικό. Πλέον, ακόμα και ο μέσος χρήστης, γνωρίζει μηχανισμούς μέσα από τους οποίους μπορεί να βρει στοιχεία για οποιοδήποτε θέμα. Κανείς όμως δε μπορεί να του εγγυηθεί επιτυχία και ταχύτητα στη διαδικασία ανεύρεσης αλλά πάνω απ' όλα, ποιότητα στα αποτελέσματα της εκάστοτε αναζήτησής του.

Συνεπώς το πρόβλημα που δημιουργείται σε πολλούς από τους χρήστες δεν είναι που θα βρουν πληροφορία, αλλά πως θα εντοπίσουν εύκολα και γρήγορα ποιοτική πληροφορία, πληροφορία που τους ενδιαφέρει και ταιριάζει με το ύφος τους. Η έλλειψη ποιότητας στης τάξεις του Διαδικτύου έχει κεντρίσει το ενδιαφέρον της επιστημονικής κοινότητας. Πολλοί ορισμοί που βρίσκονται πλέον στο επίκεντρο του ενδιαφέροντος, περιλαμβάνουν τα data mining, text analysis, text categorization, semantic web και πολλά ακόμα, τα οποία αν και ήταν γνωστά ακόμα και πριν την εξάπλωση του Διαδικτύου, φαίνονται να είναι αυτά που δίνουν λύσεις στα μειονεκτήματά του.

Στη συγκεκριμένη εργασία δε θα αναλωθούμε στην καταγραφή των πολλών, αν μη τι άλλο, προβλημάτων του Διαδικτύου αλλά θα επικεντρωθούμε σε ένα κομμάτι των προβλημάτων που προκύπτουν από την αέναη, καθημερινή και καταιγιστική δημιουργία δεδομένων και πληροφοριών. Ακόμα περισσότερο, θα εστιάσουμε την προσοχή μας στις πληροφορίες που δημιουργούνται σε καθημερινή βάση από την πληθώρα των ενημερωτικών δικτυακών πυλών που κατακλύζουν στην κυριολεξία το Διαδίκτυο. Ο λόγος για τα γνωστά *news portals*. Πρόκειται για Δικτυακούς τόπους που σαν στόχο έχουν την ενημέρωση των χρηστών του Διαδικτύου για τα φλέγοντα - κυρίως - νέα σε παγκόσμιο επίπεδο. Μερικά και πολύ σημαντικά από αυτά είναι το CNN[7], το BBC[3], το Reuters[5], το FoxNews[9], καθώς και οι υπηρεσίες που προσφέρονται από τους πολυπληθείς και από τους πλέον αναγνωρίσιμους δικτυακούς τόπους Google[13] και Yahoo[36].

Οι Δικτυακοί αυτοί τόποι εστιάζονται στο να ενημερώνουν τους χρήστες τους για ότι συμβαίνει καθημερινά στον πλανήτη. Τα νέα/άρθρα παρουσιάζονται με δομημένο τρόπο στις συγκεκριμένες σελίδες, ωστόσο το πλήθος τους είναι τέτοιο ώστε να είναι σχεδόν αδύνατο από κάποιον χρήστη να μπορέσει εντός του εικοσιτετραώρου να παρακολουθήσει όλες τις ειδήσεις που δημοσιεύονται στις πολλές διαφορετικές κατηγορίες. Ακόμα και η εστίαση σε μία συγκεκριμένη κατηγορία απαιτεί τη συνεχή και διαρκή παρακολούθηση κάθε δικτυακού τόπου προκειμένου να υπάρχει πλήρης ενημέρωση. Επίσης, πολλά από αυτά τα νέα παρουσιάζονται από την οπτική γωνία του αρθρογράφου καθώς σπάνια - πλέον - δημοσιεύονται ακέραια ακόμα και τα δελτία τύπου, με αποτέλεσμα να χάνεται συχνά το κριτήριο της αντικειμενικότητας μίας είδησης. Απόρροια όλων των παραπάνω είναι το εξής: οι χρήστες του διαδικτύου δυσκολεύονται στον εντοπισμό μίας είδησης που τους ενδιαφέρει με αποτέλεσμα να αναλώνουν το χρόνο τους στην αναζήτηση της είδησης, του νέου, του άρθρου, παρά στην ανάγνωση του ίδιου του άρθρου.

Η παρουσία των *RSS (Rich Site Summary)*, που σε ελεύθερη μετάφραση θα μπορούσαμε να καθονομάσουμε 'Περίληψη του Δικτυακού Τόπου', έρχεται να δώσει μία πρώτη λύση στο δυσβάσταχτο πρόβλημα της ανεύρεσης ενός ενδιαφέροντος άρθρου από τους αναγνώστες - χρήστες του Διαδικτύου. Η αρχή της χρήσης των RSS από τους διαχειριστές των δικτυακών τόπων φέρνει μία νέα επανάσταση και αλλάζει τα δεδομένα στην καθημερινή παγκόσμια ειδησεογραφία. Οι χρήστες έχουν ένα ακόμα κανάλι επικοινωνίας που τους προσφέρει το ελπιδοφόρο Internet. Το κανάλι είναι μία διεύθυνση - αυτή του RSS - η πρόσβαση στην οποία επιτρέπει στους χρήστες να 'έρθουν σε επαφή' με την πληροφορία που επιθυμούν και μόνον αλλά όχι με τα υπόλοιπα, άχρηστα για τους χρήστες, στοιχεία μίας ιστοσελίδας. Το μόνο που είναι απαραίτητο είναι ένα πρόγραμμα ανάγνωσης *RSS Feeds (RSS Reader)* ενώ στην πορεία ακόμα και αυτό δεν είναι αναγκαίο καθότι φυλλομετρητές του Διαδικτύου έχουν τη δυνατότητα ανάλυσης του XML εγγράφου και παρουσίασης αυτού με δομημένο και ευδιάκριτο τρόπο στους τελικούς χρήστες.

Η κατάσταση, λοιπόν, είναι η εξής: οι χρήστες έχοντας κουραστεί από την ανούσια πληροφορία που έρχεται εμπρός τους τη στιγμή που περιδιαβαίνουν έναν δικτυακό τόπο ειδησεογραφικού περιεχομένου καταφεύγουν στα RSS. Με αυτή την αλλαγή στον τρόπο 'επίσκεψης' μίας σελίδας, τα μεγάλα ειδησεογραφικά πρακτορεία παρατηρούν τη χαμηλή επισκεψιμότητα συγκεκριμένων σελίδων του δικτυακού τους τόπου οι οποίες ουσιαστικά δεν προβάλλονται προς το χρήστη ο οποίος αρκείται στο κανάλι επικοινωνίας που έχει και αποφεύγει κάθε επίσκεψη στο δικτυακό τόπο. Παράλληλα, η ανάπτυξη νέων τεχνολογιών και υπηρεσιών για το Διαδίκτυο κάνει τους διαχειριστές δικτυακών τόπων να επιθυμούν ακόμα μεγαλύτερη επισκεψιμότητα στις σελίδες τους προσφέροντας διαδραστικές υπηρεσίες, υπηρεσίες πολυμέσων κ. α. Μία κρυφή διαμάχη έχει ξεκινήσει ανάμεσα στις υπηρεσίες που 'διώχνουν' τους χρήστες από το δικτυακό τόπο και σε αυτές που τον φέρνουν ακόμα πιο κοντά σε αυτόν. Κάθε υπηρεσία προσπαθεί να υπερισχύσει της άλλης προσφέροντας ολοένα και περισσότερα στοιχεία. Το RSS έχει περιορισμένες δυνατότητες ενώ οι υπηρεσίες προσωποποιη-

μένης πρόσβασης έχουν πολλά να προσφέρουν στους χρήστες. Είναι φανερό πως οι δικτυακοί τόποι, σαν μία κανονική επιχείρηση, επιθυμούν οι χρήστες να 'έρχονται' στο δικτυακό τόπο, να επισκέπτονται όλες τις σελίδες, να βλέπουν τις διαφημίσεις, να αξιοποιούν τις νέες υπηρεσίες, να χρησιμοποιούν κάθε δεδομένο που τους προσφέρεται.

Όσο εντυπωσιακά και αν φαίνονται όλα αυτά, οι σχεδιαστές των υπηρεσιών έχουν παραλείψει σημαντικά στοιχεία. Πόσο εξοικειωμένοι είναι οι χρήστες στη χρήση περίπλοκων συστημάτων; Έχουν όλοι οι χρήστες αρκετά μεγάλη ταχύτητα στην πρόσβαση στο διαδίκτυο προκειμένου να μπορούν να χρησιμοποιούν χωρίς πρόβλημα τις προσφερόμενες υπηρεσίες; Οι χρήστες έχουν ερωτηθεί για τις πληροφορίες που θα επιθυμούσαν να τους διατίθενται; Αποτέλεσμα όλων των παραπάνω είναι: προσωποποιημένες σελίδες δικτυακών τόπων, όπου ο χρήστης αδυνατεί να τις σχεδιάσει όπως επιθυμεί καθότι 'χάνεται' στην πληθώρα δεδομένων που τους παρουσιάζονται, υπερπολλαπλασιασμός των καναλιών RSS των δικτυακών τόπων με αποτέλεσμα ο χρήστης να αντιμετωπίζει το ίδιο χάος. Τρανταχτό παράδειγμα αποτελεί το RSS Feed του CNN που αποτελείται από περισσότερα από 20 επικαλυπτόμενα κανάλια. Τέλος κάτι πολύ σημαντικό, κανείς δεν επιχειρεί να συνδυάσει τις δύο υπηρεσίες οι οποίες δε φαίνεται να διαφέρουν μεταξύ τους. Κανένας δικτυακός τόπος δεν προσπαθεί να συνδυάσει προσωποποιημένες πληροφορίες και RSS feeds.

Αποδελτιώνοντας, λοιπόν, όλα τα παραπάνω καταλήγουμε στα εξής:

- Προσωποποιημένες σελίδες: Δύσχρηστες - πολύπλοκες. Βασίζονται σε λέξεις κλειδιά ή ακόμα χειρότερα σε γενικές κατηγορίες μόνον. Ο χρήστης σε κάθε περίπτωση παραμένει εκτός της διαδικασίας κατηγοριοποίησης ή κατασκευής περίληψης που παρουσιάζεται στην προσωποποιημένη σελίδα.
- RSS feeds: Ο αριθμός τους είναι υπερβολικά μεγάλος. Ο αριθμός των άρθρων που περιέχουν είναι υπερβολικά μεγάλος. Συνήθως δε χρησιμοποιούνται σωστά.

Όλα τα παραπάνω έχουν ως αποτέλεσμα οι χρήστες να δυσκολεύονται στην αναζήτηση ειδήσεων και πιο συγκεκριμένα, στην παρακολούθηση αποκλειστικά των ειδήσεων που τους ενδιαφέρουν. Ακόμα περισσότερο, οι χρήστες θα πρέπει με κάποιον τρόπο να γίνουν κομμάτι του πυρήνα ενός τέτοιου συστήματος και να διαμορφώνουν τον τρόπο με τον οποίο πραγματοποιείται η κατηγοριοποίηση αλλά και τον τρόπο με τον οποίο παρουσιάζονται τα αποτελέσματα της αναζήτησης σε αυτούς.

Στη συνέχεια θα παρακολουθήσουμε κάθε διαδικασία του συστήματος που παρουσιάζεται και θα αναλυθούν τα προβλήματα που εντοπίζονται σε καθέ μια από αυτές. Το σύστημά μας ακολουθεί μία διαδικασία σειριακά προκειμένου να παράγει το ζητούμενο αποτέλεσμα το οποίο είναι η παρουσίαση προσωποποιημένων, κατηγοριοποιημένων άρθρων στον τελικό χρήστη. Για να γίνει αυτό θα πρέπει το σύστημα να είναι σε θέση να συλλέγει συνεχώς άρθρα από μεγάλα ειδησεογραφικά πρακτορεία. Η συλλογή των άρθρων δεν είναι αρκετή. Αφού τα άρθρα συγκεντρωθούν, θα πρέπει να εφαρμοστούν σε αυτά μία σειρά από αλγορίθμους προκειμένου να 'καθαριστεί' το κείμενό τους από οποιαδήποτε περιττή πληροφορία. Εν συνεχεία θα πρέπει να εφαρμοστούν αλγόριθμοι κατηγοριοποίησης του κειμένου και εξαγωγής περίληψης. Τέλος θα πρέπει να υπάρχει ένας μηχανισμός ο οποίος θα πραγματοποιεί προσωποποίηση των πιο πρόσφατων άρθρων στον εκάστοτε χρήστη και φυσικά ένα σύστημα παρουσίασης της πληροφορίας στον τελικό χρήστη το οποίο θα προσαρμόζει το περιεχόμενο στις δυνατότητες της εκάστοτε συσκευής.

1.2 Ιστορική αναδρομή

Η διαδικασία δημιουργίας αποδοτικής, αυτόματα εξαγόμενης περίληψης κειμένου έχει τις αρχές της στα τέλη της δεκαετίας του 1950 με την αναλυτική προσέγγιση από τον H. P. Luhn [89]. Η κλασική αυτή εργασία βασίζεται στην ανάλυση των λέξεων και των προτάσεων που απαρτίζουν ένα κείμενο. Κάποιες τεχνικές εισάγουν την έννοια της αναζήτησης ιδιαίτερων λέξεων και φράσεων στο κείμενο, ενώ άλλες βασίζονται σε πρότυπα συσχέτισης μεταξύ προτάσεων, ή λαμβάνουν υπ' όψιν τους το μέγεθος των προτάσεων. Πιο εξειδικευμένες τεχνικές, δεν χρησιμοποιούν στοιχεία από τη συλλογή κειμένων αλλά επιχειρούν να παράγουν την περίληψη απευθείας χρησιμοποιώντας μια βασισμένη στη γνώση αναπαράσταση του περιεχομένου ή ενός στατιστικού μοντέλου του κειμένου. Ακόμη, ερευνώνται πιθανοτικά μοντέλα κατανομής όρων στα κείμενα.

Σε γενικές γραμμές, οι τεχνικές περίληψης κειμένου μπορούν να χωριστούν σε τέσσερις βασικές κατηγορίες:

- Ευρετικές

- TF-IDF
- Βασισμένες στη γνώση
- Βασισμένες σε στατιστικά μοντέλα

Ένας άλλος τρόπος κατηγοριοποίησης των τεχνικών περίληψης, εισάγεται από τους Mani [90] και Hahn [70] σχετικά με τη εμπλοκή της γνώσης για το πεδίο. Οι δύο κατηγορίες που προτείνονται είναι η 'πλούσια σε γνώση' και η 'φτωχή σε γνώση'. Η πρώτη περιλαμβάνει μεθόδους οι οποίες δεν λαμβάνουν υπ' όψιν τους την γνώση που μπορεί να υπάρχει για το πεδίο γνώσης του συγκεκριμένου κειμένου, ενώ η δεύτερη κατηγορία περιλαμβάνει μεθόδους που λαμβάνουν αυτό τον παράγοντα υπ' όψιν τους στην προσπάθεια τους να παράγουν καλύτερες περιλήψεις. Σύμφωνα με αυτού του είδους την κατηγοριοποίηση μπορούμε να πούμε ότι οι ευρετικές και οι TF-IDF μέθοδοι είναι 'φτωχές σε γνώση', ενώ οι βασιζόμενες σε γνώση και οι στατιστικών μοντέλων είναι μέθοδοι 'πλούσιες σε γνώση'

1.3 Περιγραφή της εργασίας

Βασιζόμενοι στα παραπάνω, στοχεύουμε στο να αναπτύξουμε ένα έξυπνο σύστημα το οποίο θα έχει ως βασικό σκοπό την παροχή ποιοτικού περιεχομένου στους χρήστες του. Η παροχή ποιοτικού περιεχομένου δε μπορεί να βασίζεται αποκλειστικά και μόνο στη δημιουργία περίληψης κειμένων, γιατί μία τέτοια λύση το μόνο που θα μπορούσε να προφέρει είναι επιπλέον περιεχόμενο στο ήδη χαοτικό κόσμο του Διαδικτύου. Στοχεύουμε λοιπόν στην ανάπτυξη μίας ολόκληρης πλατφόρμας.

Η πλατφόρμα που σκοπεύουμε να αναπτύξουμε θα πρέπει να περνά την πληροφορία μέσα από πολλά στάδια επεξεργασίας προτού αυτή καταλήξει στον τελικό αποδέκτη της. Μέσα από τη χρήση τεχνολογιών διαδικτύου αλλά και καθαρού μαθηματικού λογισμού, θα προσπαθήσουμε να εφαρμόσουμε κατάλληλους αλγορίθμους στην πληροφορία, προκειμένου να πετύχουμε το βέλτιστο 'φιλτράρισμα' αφήνοντας το χρήστη να παραλάβει αποκλειστικά και μόνο χρήσιμη και επιθυμητή πληροφορία. Αυτό θα γίνει μέσα από μία σειρά συστημάτων που θα λειτουργούν παράλληλα και θα συνεργάζονται προκειμένου να πετύχουν το επιθυμητό αποτέλεσμα. Η πλατφόρμα δε σταματά όμως σε αυτό το σημείο. Άλλωστε πολλές έρευνες έχουν γίνει πάνω στο συγκεκριμένο θέμα και αρκετοί είναι οι αλγόριθμοι φιλτραρίσματος πληροφορίας που έχουν προταθεί από την επιστημονική κοινότητα. Το σύστημα θα μπορεί να παρέχει περίληψη της πληροφορίας εισόδου στον χρήστη, κατηγοριοποιημένη και προσωποποιημένη για αυτόν και στην μορφή που επιθυμεί, υποστηρίζοντας έτσι κάθε είδους συσκευή απεικόνισης από την μεριά του χρήστη (π. χ. συσκευές μικρού μήκους).

1.4 Δομή της εργασίας

Η υπόλοιπη εργασία δομείται ως εξής: στο κεφάλαιο 2 παρουσιάζονται τα θέματα που θα μας απασχολήσουν καθώς και οι τρέχουσες εξελίξεις στα ερευνητικά πεδία (State of the Art). Στο κεφάλαιο 3 παρουσιάζεται το κομμάτι των σχετικών εργασιών, ενώ στο κεφάλαιο 4 γίνεται μια γενικότερη περιγραφή της αρχιτεκτονικής και των χαρακτηριστικών του συστήματος που αναπτύχθηκε. Ακολουθεί η αλγοριθμική περιγραφή του μηχανισμού του συστήματος (κεφάλαιο 5) και η παρουσίαση της βάσης δεδομένων που χρησιμοποιήθηκε (κεφάλαιο 6). Στο κεφάλαιο 7 γίνεται μια συνοπτική παρουσίαση των διαθέσιμων τεχνολογιών υλοποίησης καθώς και των επιλογών που έγιναν για τα διάφορα υποσυστήματα του μηχανισμού. Ακολουθεί, στο κεφάλαιο 8, η παρουσίαση της υλοποίησης του συστήματος και στο κεφάλαιο 9 οι προδιαγραφές και η χρήση του. Στο κεφάλαιο 10 παρουσιάζονται τα πειραματικά αποτελέσματα καθώς και η αξιολόγησή του συστήματος. Τέλος στο κεφάλαιο 11 δίνονται τα συμπεράσματα που προκύπτουν από την εργασία καθώς και προτάσεις για μελλοντική επέκτασή της.

ΚΕΦΑΛΑΙΟ 2

Τα θέματα που θα μας απασχολήσουν

Science is a differential equation.
Religion is a boundary condition.

Alan Turing, English logician & mathematician

Στο συγκεκριμένο κεφάλαιο παρουσιάζονται τα ερευνητικά θέματα με τα οποία καταπιάνεται η εργασία. Εντοπίζονται τα τεχνολογικά προβλήματα που υπάρχουν, τρόποι με τους οποίους έχουν αντιμετωπισθεί καθώς και η δική μας προσέγγιση. Οι παράγραφοι που ακολουθούν δίνουν μια αναλυτική παρουσίαση του state of the art στα θέματα της ανάκτησης πληροφορίας και εξόρυξης δεδομένων.

2.1 Σημασιολογικός Ιστός και Μεταδεδομένα

Το Διαδίκτυο σήμερα αποτελεί τη μεγαλύτερη πηγή πληροφοριών. Μεγάλοι όγκοι δεδομένων αναζητούνται, ανταλλάσσονται και επεξεργάζονται μέσω του Παγκοσμίου Ιστού. Επειδή όμως ο όγκος των δεδομένων του Ιστού έχει πάρει μεγάλες διαστάσεις, χωρίς να υπάρχει ενιαίος τρόπος οργάνωσης, η ανταλλαγή και η επεξεργασία τους είναι πολύ δύσκολη. Ο *Σημασιολογικός Ιστός* έρχεται ακριβώς να εξυπηρετήσει την ανάγκη για ενιαία οργάνωση των δεδομένων, ώστε το Διαδίκτυο να γίνει μια αποδοτική παγκόσμια πλατφόρμα ανταλλαγής και επεξεργασίας πληροφορίας από ετερογενείς πηγές. Ένας γενικός ορισμός μας λέει ότι ο Σημασιολογικός Ιστός δίνει δομή, οργάνωση και σημασιολογία στα δεδομένα, ώστε να είναι, σε μεγάλο βαθμό, κατανοητά από μηχανές (machine understandable).

Ο όρος Σημασιολογικός Ιστός (Semantic Web) χρησιμοποιήθηκε για πρώτη φορά το 1998 από το δημιουργό του πρώτου φυλλομετρητή ιστοσελίδων και εξυπηρετητή διαδικτύου, Tim Berners-Lee [44]. Από τότε καταβάλλεται μεγάλη προσπάθεια από την επιστημονική κοινότητα για την υλοποίησή του πάνω από τον Παγκόσμιο Ιστό. Στο βασικότερο επίπεδό του, ο Σημασιολογικός Ιστός αποτελεί μία συλλογή από συνοπτική πληροφορία για τη διακινούμενη πληροφορία, τα μεταδεδομένα, η οποία δεν είναι ορατή στον τελικό χρήστη. Τα μεταδεδομένα χρησιμοποιούνται για να περιγράψουν υπάρχοντα έγγραφα, ιστοσελίδες, βάσεις δεδομένων, προγράμματα που βρίσκονται στο διαδίκτυο. Οι εφαρμογές λογισμικού που κάνουν χρήση μεταδεδομένων αποκτούν καλύτερη κατανόηση της σημασιολογίας του περιεχομένου τους και άρα μπορούν να τα επεξεργαστούν με πιο αποδοτικό τρόπο. Η κατανόηση των μεταδεδομένων από τις μηχανές είναι δυνατή μέσω της χρήσης ειδικών λεξικών (των οντολογιών) τα οποία παρέχουν κοινούς κανόνες και λεξιλόγια για την ερμηνεία των δεδομένων. Με αυτό τον τρόπο είναι δυνατή η κοινή κατανόηση όρων και εννοιών από εφαρμογές που προέρχονται από διαφορετικά πληροφοριακά συστήματα. Απώτερος στόχος της όλης προσπάθειας είναι η ικανοποίηση των απαιτήσεων των συμμετεχόντων στην Κοινωνία της Πληροφορίας για αυξημένη ποιότητα υπηρεσιών. Αυτό συνίσταται κυρίως στη βελτιωμένη αναζήτηση, εκτέλεση σύνθετων διεργασιών μέσω του Διαδικτύου και στην εξατομίκευση της πληροφορίας σύμφωνα με τις ανάγκες του εκάστοτε χρήστη.

Ένα από τα σημαντικότερα προβλήματα που καλείται να λύσει ο Σημασιολογικός Ιστός είναι η πρόσβαση στην πληροφορία. Σύμφωνα με πρόσφατες μελέτες, η ανθρωπότητα έχει παράγει από το 1999 μέχρι το 2003, τόσες νέες πληροφορίες όσες παρήγαγε όλα τα προηγούμενα χρόνια της ιστορίας της. Σε αυτό το διάστημα των τριών τελευταίων ετών παρήχθησαν 12 exabytes πληροφορίας υπό τη μορφή έντυπου, οπτικού ή και ηχητικού υλικού. Η αυξανόμενη αυτή παραγωγή και η συνεχής βελτίωση των μεθόδων ψηφιοποίησης συμβάλλουν στην παραγωγή ενός ωκεανού ψηφιακών δεδομένων που προφανώς δύναται να δημιουργήσει μεγάλο αριθμό προβλημάτων. Το πιο σημαντικό ίσως από αυτά είναι ο τρόπος με τον οποίο θα μπορεί κανείς να διαχειριστεί όλη αυτή την πληροφορία. Δε θα πρέπει φυσικά να αμελούμε το γεγονός πως η ικανότητα παραγωγής, αποθήκευσης και μετάδοσης της πληροφορίας έχει ξεπεράσει κατά πολύ τις δυνατότητες αναζήτησης, πρόσβασης και παρουσίασης.

Λόγω του αυξανόμενου όγκου της πληροφορίας και των προβλημάτων αποτελεσματικής πρόσβασης, έχει γίνει τα τελευταία χρόνια ξεκάθαρο προς την επιστημονική κοινότητα ότι για την αύξηση της απόδοσης, χρειάζονται νέες μέθοδοι υπολογισμού ικανές να προσαρμοστούν σε μία πληθώρα παραμέτρων τόσο αντικειμενικών όσο και υποκειμενικών. Η απόδοση ενός συστήματος πρόσβασης στην πληροφορία εκτιμάται μέσα από την ανάκληση και την ακρίβεια που διαθέτει.

Η αναφορά στα προβλήματα που αντιμετωπίζουν τα σύγχρονα συστήματα πρόσβασης στην πληροφορία έχει άμεση σχέση με τον τύπο των ερωτήσεων που δέχονται ως είσοδο. Υπάρχουν δύο διαφορετικά είδη ερωτημάτων, οι ερωτήσεις γενικού περιεχομένου και ειδικού περιεχομένου. Το μέγεθος της απάντησης σε ερωτήσεις γενικού περιεχομένου είναι μεγάλο και παρουσιάζει εξαιρετικά μεγάλες αποκλίσεις ως προς τη σχετικότητα της ίδιας της ερώτησης. Το πρόβλημα εστιάζεται στην επιλογή ενός μικρού συνόλου από τις πιο σχετικές απαντήσεις, είναι δηλαδή πρόβλημα ακρίβειας. Αντίθετα, για τις ερωτήσεις ειδικού περιεχομένου, το διαθέσιμο σύνολο σχετικών απαντήσεων είναι μικρό και το πρόβλημα που προκύπτει είναι πρόβλημα ανάκτησης.

Εκτός από τα κλασσικά προβλήματα που αντιμετωπίζουν τα ΠΣ στον τομέα της πρόσβασης στην πληροφορία, αναδύονται και άλλα άμεσα συνδεδεμένα με το είδος της ίδιας της πληροφορίας:

- Συνωνυμία: ανάκτηση μη σχετικών απαντήσεων που περιέχουν όρους συνώνυμους με αυτούς της ερώτησης.
- Ασάφεια / Διφορούμενες έννοιες: ανάκτηση μη σχετικών αποτελεσμάτων λόγω ασάφειας της ερώτησης ή λόγω ύπαρξης διφορούμενων εννοιών.
- Πειθώ των μηχανών αναζήτησης *(search engine persuasion)*:ταξινόμηση των ανακτημένων εγγράφων με βάση το βαθμό σχετικότητας τους προς την ερώτηση έχοντας υπόψη τα προβλήματα της συνωνυμίας και της ασάφειας.

Τα τελευταία χρόνια, μια νέα ερευνητική προσπάθεια έχει επικεντρωθεί σε αυτό το πεδίο το οποίο ανήκει στην περιοχή που ονομάζεται Προσαρμοσμένη Πρόσβαση στην Πληροφορία *(Adaptive Information Access)*. Η πρόσβαση στην πληροφορία αφορά αρκετές ερευνητικές περιοχές που θα μπορούσαν να συνδυαστούν για την κατασκευή συστημάτων ικανών να ανταποκριθούν στις σύγχρονες ανάγκες. Τέτοιες περιοχές είναι η έξυπνη αναζήτηση πληροφορίας, μάθηση μηχανής και αλληλεπίδραση ανθρώπου υπολογιστή. Στην παρούσα διπλωματική θα ασχοληθούμε με ζητήματα που έχουν να κάνουν τόσο με έξυπνη ανάκτηση πληροφορίας, με μηχανική μάθηση όσο και με αλληλεπίδραση χρηστών με τον υπολογιστή.

2.2 Εξόρυξη πληροφορίας από το Διαδίκτυο

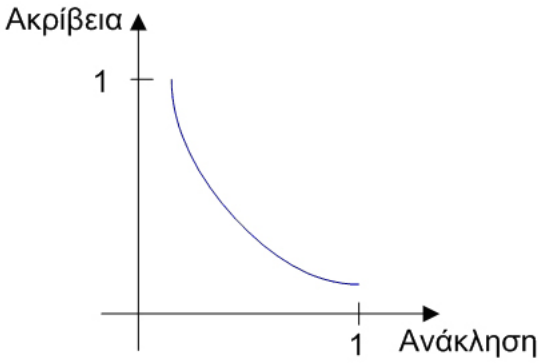
Εξόρυξη πληροφορίας από το Διαδίκτυο ονομάζεται κάθε διαδικασία που έχει σαν αποτέλεσμα ανάκτηση πληροφορίας *(Information Retrieval)* από τον παγκόσμιο ιστό. Στο εξής θα αναφερόμαστε στον όρο ανάκτηση πληροφορίας ως IR για συντομία. Η ανακτώμενη πληροφορία δεν περιορίζεται απλώς σε σελίδες HTML, αλλά μπορεί να αφορά και αρχεία πολυμέσων ή οποιοδήποτε είδος αρχείου μπορεί να μεταφερθεί πάνω από το Διαδίκτυο. Η ανάγκη για ανάκτηση πληροφορίας πηγάζει από τις αρχές της δεκαετίας του 50 όταν ο Mooers [101] εξέφρασε ανοιχτά σε δημοσίευσή του αυτή την ανάγκη. Αργότερα, στη δεκαετία του 60, το IR είχε γίνει πλέον ένα πολύ δημοφιλές θέμα καθώς πολλοί ερευνητές πίστευαν ότι μπορούν να αυτοματοποιήσουν τις μέχρι τότε χειροκίνητες διαδικασίες όπως η δεικτοδότηση και η αναζήτηση.

Προκειμένου να πετύχει το στόχο της, η κοινότητα IR όρισε δύο βασικές ενέργειες που έχουν γίνει αντικείμενα έρευνας για πολλά χρόνια και είναι: η δεικτοδότηση και η αναζήτηση. Η δεικτοδότηση αναφέρεται στον τρόπο με τον οποίο αναπαρίσταται η πληροφορία για τους σκοπούς της ανάκτησης. Η αναζήτηση αναφέρεται στον τρόπο με τον οποίο δομείται η πληροφορία όταν πραγματοποιείται ένα ερώτημα. Παρόλο που οι δύο αυτές διαδικασίες αποτελούν τον πυρήνα ενός συστήματος IR, άλλες διαδικασίες κερδίζουν επίσης έδαφος, όπως οι τεχνικές αναπαράστασης της πληροφορίας, με σκοπό να βελτιωθεί η αποτελεσματικότητα της ανάκτησης.

Στην παρούσα φάση το IR αντιμετωπίζει μία σειρά από θέματα. Αρχικά, εφαρμόστηκε σε ΒΔ βιβλιοθηκών, όπου σε ένα αρχείο αποθηκεύονταν γενικά χαρακτηριστικά κάθε εγγράφου, όπως ο τίτλος και ο συγγραφέας, και η αναζήτηση γινόταν βάσει αυτών των στοιχείων. Στη συνέχεια, και εξ' αιτίας της αύξησης του μεγέθους των αποθηκευτικών μέσων, ολόκληρο το κείμενο αποθηκευόταν σε αρχείο και η αναζήτηση ήταν εφικτή σε ολόκληρες συλλογές από κείμενα. Έτσι μέχρι ενός σημείου το IR αντιπροσώπευε την ανάκτηση κειμένων. Αργότερα και έως σήμερα, δίνεται περισσότερη σημασία στον όρο πληροφορία (Information). Άλλωστε σήμερα δεν έχουμε μόνο έγγραφα πάνω στα οποία γίνεται η αναζήτηση αλλά και αρχεία πολυμέσων. Ωστόσο το βασικό κλειδί στην υπόθεση του IR είναι ανάκτηση κειμένων ή πληροφορίας που προσεγγίζουν περισσότερο τις ανάγκες του χρήστη που πραγματοποιεί την αναζήτηση.

Ένα από τα βασικά στοιχεία του IR είναι η μέτρηση του κατά πόσο τα ανακτημένα κείμενα είναι σχετικά με το ερώτημα που κάνουμε. Έτσι λοιπόν, ένα βασικό στοιχείο στο οποίο εστιάζουμε είναι η εύρεση μετρικών που θα μπορούν να αναπαραστήσουν αριθμητικά τη σχετικότητα των αποτελεσμάτων ενός συστήματος IR. Πολλές μετρικές έχουν αναπτυχθεί με τις δύο πιο γνωστές να είναι η ανάκληση και η ακρίβεια. Η ακρίβεια μας δίνει το ποσοστό (%) των σχετικών κειμένων εν συγκρίσει με αυτά που ανακτήθηκαν ενώ η ανάκληση μας δίνει το ποσοστό (%) των κειμένων που ανακτήθηκαν εν συγκρίσει με μία συλλογή που γνωρίζουμε ότι περιέχει όλα τα σχετικά.

Η συνηθισμένη απόκριση που έχει ένα σύστημα IR είναι αυτή που φαίνεται στο παρακάτω σχήμα (Σχήμα 2.1) στο οποίο φαίνεται ότι τα μεγέθη *ακρίβεια και ανάκληση* είναι αντιστρόφως ανάλογα. Αυτό σημαίνει πως για αν αυξήσουμε την ανάκληση θα μειωθεί η ακρίβεια. Φυσικά ισχύει και το αντίστροφο.



Σχήμα 2.1: Σχεδιάγραμμα ακρίβειας - ανάκλησης.

2.2.1 Ανάκτηση πληροφορίας και φιλτράρισμα πληροφορίας

Ένα σύστημα IR μπορεί να πετύχει κατά μέσο όρο περίπου 30% ανάκληση και 30% ακρίβεια. Οι τιμές αυτές δεν έχουν καμία σύγκριση με ένα σύστημα DBMS που τα ποσοστά αυτά προσεγγίζουν το 100%. Ωστόσο θα μπορούσε κανείς να πει πως και τα δύο συστήματα πραγματοποιούν την ίδια διαδικασία, δηλαδή ανάκτηση πληροφορίας. Αυτό βέβαια έχει να κάνει με τον τρόπο με τον οποίο δομείται ένα σύστημα DBMS και ο οποίος είναι τέτοιος ώστε να εξυπηρετεί απόλυτα τις ανάγκες ενός χρήστη.

Αυτή η δυσκολία που αντιμετωπίζουν τα συστήματα IR (μικρές τιμές ανάκλησης και ακρίβειας) γεννούν ένα άλλο επιστημονικό πεδίο το οποίο υπάρχει παράλληλα με το IR και είναι το *IF (Information Filtering)*. Σε ένα κλασσικό άρθρο οι Belkin και Croft παρουσίασαν δύο διαφορετικούς ορισμούς για τα δύο παραπάνω θέματα οι οποίοι έχουν κοινές τεχνικές αλλά διαφέρουν σε τρία βασικά στοιχεία [41]. Πρώτον, στο IR

όταν ο χρήστης κάνει ένα ερώτημα περιμένει άμεση απόκριση. Στο IF ο χρήστης μπορεί να περιμένει, εν γνώσει του, για μεγάλο χρονικό διάστημα μέχρι να του παρουσιαστεί μία απάντηση. Επιπρόσθετα το IF χειρίζεται και θέματα που από τη φύση τους είναι δυναμικά και εντάσσει στο μηχανισμού του στοιχεία εκμάθησης σύμφωνα με τα κείμενα που προσθέτει στη συλλογή του. Τέλος, το βασικότερο είναι πως το IR αναζητά παραπλήσια κείμενα από μία μεγάλη συλλογή κειμένων σε αντίθεση με το IF το οποίο προσπαθεί να αφαιρέσει από μία συλλογή τα εισερχόμενα κείμενα που δεν είναι σχετικά.

Παρ' όλες τις διαφορές που έχουν τα δύο αυτά πεδία δεν πρέπει να αμελούμε πως έχουν παραπλήσιο σκοπό: να εξασφαλίσουν ότι τα κείμενα που θα παρουσιαστούν στο χρήστη είναι σχετικά με το ερώτημά του.

Τα διαγράμματα ακρίβειας/ανάκλησης είναι χρήσιμα εφόσον μελετούμε την απόδοση ανάκτησης διαφορετικών αλγορίθμων σε ένα σύνολο από πρότυπες πληροφοριακές ανάγκες. Ωστόσο υπάρχουν περιπτώσεις στις οποίες θα θέλαμε να συγκρίνουμε την απόδοση αλγορίθμων ανάκτησης για ατομικές πληροφοριακές ανάγκες. Οι λόγοι για να το κάνουμε αυτό είναι δύο:

1. η χρήση μέσων τιμών που προκύπτουν από την εκτέλεση διαφόρων ερωτημάτων μπορεί να αποκρύπτει σημαντικές ανωμαλίες στον αλγόριθμο ανάκτησης,

2. όταν συγκρίνουμε δύο αλγορίθμους, μπορεί να θέλουμε να μελετήσουμε κατά πόσο ο ένας είναι καλύτερος του άλλου για κάθε μία από τις πληροφοριακές ανάγκες που έχουμε και όχι συνολικά.

Σε τέτοιες περιπτώσεις υπολογίζουμε μία μόνο τιμή ακρίβειας για κάθε ερώτημα, η οποία θα μπορούσε να θεωρηθεί σαν σύνοψη του συνολικού διαγράμματος ακρίβειας/ανάκλησης. Συνήθως αυτή η τιμή είναι η ακρίβεια σε κάποιο συγκεκριμένο επίπεδο ανάκλησης. Φυσικά αυτές είναι λίγες από τις πολλές προσεγγίσεις που μπορούν να γίνουν.

2.2.2 Μοντέλα ανάκτησης πληροφορίας

Τα τρία κλασσικά μοντέλα στην Ανάκτηση Πληροφορίας είναι το *Boolean*, το *Vector Space* και το Πιθανοτικό. Στο μοντέλο Boolean, τόσο τα κείμενα όσο και τα ερωτήματα αντιμετωπίζονται ως ένα σύνολο από όρους δεικτοδότησης. Κατά συνέπεια το μοντέλο μπορεί να θεωρηθεί ως συνολοθεωρητικό. Στο Vector Space, τα κείμενα και τα ερωτήματα αναπαρίστανται ως διανύσματα σε έναν $t-$διάστατο χώρο. Έτσι λέμε ότι το μοντέλο είναι αλγεβρικό. Το Πιθανοτικό μοντέλο εισάγει έναν τρόπο αναπαράστασης, ο οποίος βασίζεται στην πιθανοθεωρία και κατά συνέπεια το μοντέλο είναι πιθανοτικού χαρακτήρα. Με τον καιρό προτάθηκαν διάφορες νέες προσεγγίσεις σε καθεμιά από τις κατηγορίες βασικών μοντέλων. Έτσι έχουμε στο συνολοθεωρητικό πεδίο τα μοντέλα, ασαφές (fuzzy) Boolean και επεκτεταμένο Boolean. Στα αλγεβρικά μοντέλα έχουμε το γενικευμένο vector space, την λανθάνουσας σημασιολογικής δεικτοδότησης (LSI) και το μοντέλο των νευρωνικών δικτύων. Στον πιθανοτικό τομέα εμφανίστηκαν τα δίκτυα εξαγωγής συμπεράσματος (inference networks) και τα δίκτυα πεποίθησης (belief networks). Εκτός από την χρήση του περιεχομένου των κειμένων, ορισμένα μοντέλα εκμεταλλεύονται και την εσωτερική δομή που φυσιολογικά υπάρχει στο γραπτό λόγο. Σε αυτή την περίπτωση λέμε ότι έχουμε ένα δομημένο μοντέλο. Για τη δομημένη ανάκτηση κειμένου, συναντούμε δύο μοντέλα, τις μη επικαλυπτόμενες λίστες (non-overlapping lists) και τους κοντινούς κόμβους *(proximal nodes)*.

Τυπικός ορισμός των μοντέλων

Πριν προχωρήσουμε στην εξέταση των επί μέρους μοντέλων θα δώσουμε έναν τυπικό και ακριβή ορισμό για το τι είναι ένα μοντέλο ΑΠ.

Ορισμός 2.2.1. Ένα μοντέλο ανάκτησης πληροφορίας είναι η τετράδα $[D, Q, F, R(qi, dj)]$ όπου:

1. D είναι ένα σύνολο από λογικές αναπαραστάσεις για τα κείμενα της συλλογής

2. Q είναι ένα σύνολο από λογικές αναπαραστάσεις για τις πληροφοριακές ανάγκες του χρήστη. Αυτές οι αναπαραστάσεις καλούνται ερωτήματα

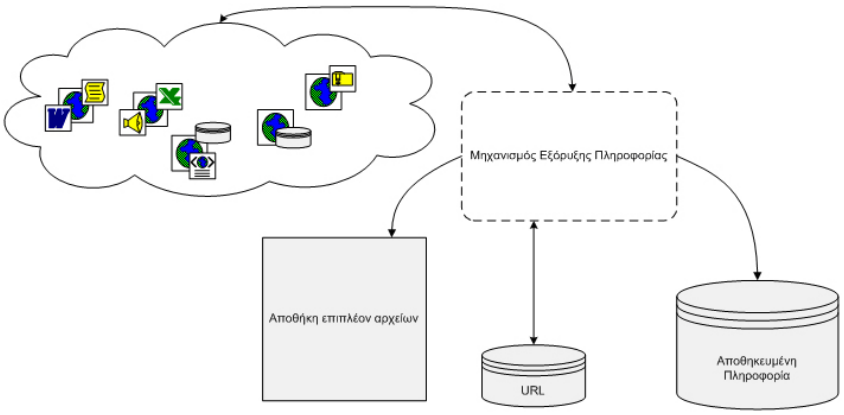
3. F είναι ένα υπόβαθρο για την μοντελοποίηση της αναπαράστασης των κειμένων, των ερωτημάτων και των σχέσεων μεταξύ τους

4. $R(q_i, d_j)$ είναι μια συνάρτηση κατάταξης, η οποία συνδέει έναν πραγματικό αριθμό με ένα ερώτημα $q_i \in Q$ και μια αναπαράσταση κειμένου $d_j \in D$. Μια τέτοια κατάταξη ορίζει μια διάταξη πάνω στα κείμενα πάντα με βάση το ερώτημα q_i.

Διαισθητικά ο παραπάνω ορισμός περιγράφει τη διαδικασία καθορισμού ενός μοντέλου ΑΠ. Η διαδικασία ορισμού ενός μοντέλου είναι η ακόλουθη. Αρχικά επινοείται ένας τρόπος αναπαράστασης για τα κείμενα και την πληροφοριακή ανάγκη του χρήστη. Έπειτα καθορίζεται ένα υπόβαθρο στο οποίο θα μπορούν αυτές οι αναπαραστάσεις να μοντελοποιηθούν. Το υπόβαθρο αυτό, θα πρέπει να μπορεί να παρέχει και τον μηχανισμό κατάταξης. Για παράδειγμα στο *Boolean* μοντέλο, το υπόβαθρο αυτό αποτελείται από τις αναπαραστάσεις των κειμένων και των ερωτήσεων ως σύνολα, και τις κλασσικές πράξεις πάνω στα σύνολα. Αντίστοιχα στο Vector space, το υπόβαθρο αποτελείται από τις διανυσματικές αναπαραστάσεις κειμένων στον $t-$διάστατο διανυσματικό χώρο και τις επιτρεπτές αλγεβρικές πράξεις πάνω σε διανύσματα.

2.2.3 Αρχιτεκτονική μηχανισμών εξόρυξης

Όλες οι μηχανές αναζήτησης πραγματοποιούν ανάκτηση πληροφορίας προκειμένου να μπορούν να εξυπηρετούν τους χρήστες τους. Έτσι, μέχρι σήμερα έχει κατασκευαστεί πληθώρα προγραμμάτων τα οποία είτε λειτουργώντας σαν αυτόνομες μονάδες είτε σε συνεργασία μεταξύ τους πραγματοποιούν εξόρυξη πληροφορίας. Η γενική ιδέα ενός μηχανισμού εξόρυξης πληροφορίας είναι εξαιρετικά απλή και φαίνεται στο παρακάτω σχήμα (Σχήμα 2.2).



Σχήμα 2.2: Μηχανισμός Εξόρυξης Πληροφορίας.

Ένας τέτοιος μηχανισμός μπορεί να είναι ένας απλός υπολογιστής ή ακόμα και μερικές χιλιάδες υπολογιστές που λειτουργούν κάτω από την επίβλεψη ενός. Ο μηχανισμός ξεκινά να λειτουργεί περιδιαβαίνοντας σελίδες του Διαδικτύου. Οι HTML σελίδες αποθηκεύονται σε μία βάση δεδομένων μαζί με επιπρόσθετες πληροφορίες για αυτές οι οποίες μπορεί να περιλαμβάνουν: το URL, την ώρα που ανακτήθηκε η σελίδα, το μέγεθός της και άλλα. Σε μία ξεχωριστή (συνήθως) βάση δεδομένων αποθηκεύονται όλα τα URL που έχουν ανακτηθεί και τα οποία ανακτώνται ανά τακτά χρονικά διαστήματα. Παράλληλα κάθε σελίδα αναλύεται προκειμένου να εξαχθούν από αυτή όλα τα links που περιέχει (σύμβολο $< a >$ στην HTML). Τα links που 'διαβάζει' ο μηχανισμός συγκρίνονται με αυτά που υπάρχουν αποθηκευμένα στη βάση δεδομένων URL και γίνονται οι κατάλληλες προσθήκες. Τέλος, κάποια επιπλέον αρχεία (doc, css, xml, scripts, πολυμέσα) αποθηκεύονται συνήθως σε καταλόγους που ονομάζονται κατάλληλα από τον μηχανισμό, έτσι ώστε να είναι σε θέση να τα προσπελάσει ανά πάσα στιγμή.

Μερικοί από τους πιο γνωστούς μηχανισμούς που πραγματοποιούν εξόρυξη πληροφορίας είναι οι *crawlers*, τα *bots*, τα *spiders* κ. α. Η λειτουργία τους είναι ουσιαστικά ίδια και βασίζεται στην αρχιτεκτονική που περιγράφηκε στο παραπάνω σχήμα.

2.2.4 Τεχνολογίες ανάκτησης δεδομένων από το Διαδίκτυο

Η ανάκτηση πληροφορίας είναι μία έννοια η οποία αναφέρεται σε κάθε μηχανισμό ο οποίος μέσω ενός αλγορίθμου 'επιστρέφει' αποτελέσματα από ένα σύνολο στοιχείων. Μιλώντας για ανάκτηση πληροφορίας από το διαδίκτυο θα πρέπει να αναλογιστούμε τη μοναδικότητα των στοιχείων που χαρακτηρίζουν το Διαδίκτυο και συνεπώς αλλάζουν τη διαδικασία ανάκτησης δεδομένων από αυτό. Τα κύρια χαρακτηριστικά του Διαδικτύου είναι:

- Εξαιρετικά μεγάλο μέγεθος
  - Σύμφωνα με πρόσφατους υπολογισμούς το μέγεθος του Διαδικτύου ξεπερνά τις 11 δισεκατομμύρια σελίδες [35].
- Δυναμικός χαρακτήρας
  - Το Internet αλλάζει ώρα με τη ώρα, ενώ στα κλασσικά συστήματα ανάκτησης δεδομένων υπάρχουν σταθερές βάσεις δεδομένων.
- Περιέχει ετερογενές υλικό
  - Υπάρχουν πολλοί διαφορετικοί τύποι αρχείων (κείμενα, εικόνες, βίντεο, ήχος, scripts) με αποτέλεσμα οι αλγόριθμοι ανάκτησης δεδομένων να πρέπει να εφαρμοστούν τόσο σε απλό κείμενο όσο και πολυμεσικά δεδομένα.
- Υπάρχει μεγάλο εύρος γλωσσών
  - Οι γλώσσες που χρησιμοποιούνται στο Διαδίκτυο υπολογίζονται σε πάνω από 100.
- Διπλές εγγραφές
  - Η αντιγραφή είναι ένα βασικό χαρακτηριστικό ου Διαδικτύου. Δεν είναι τυχαίο πως 25-30% των σελίδων του Διαδικτύου αποτελούν αντίγραφα άλλων σελίδων.
- Πολλά links από μία σελίδα σε άλλες
  - Υπολογίζεται πως σε κάθε σελίδα περιέχονται κατά μέσο όρο 10 links προς άλλες σελίδες.
- Πολλοί και διαφορετικών ειδών χρήστες
  - Κάθε χρήστης έχει τα δικές του ανάγκες αλλά και τις δικές του γνώσεις και απαιτήσεις από το Διαδίκτυο.
- Διαφορετική συμπεριφορά από τους χρήστες
  - Έχει υπολογιστεί πως περίπου το 90% των χρηστών του Διαδικτύου παρατηρούν μόνο την πρώτη σελίδα από αυτές που του επιστρέφει μία μηχανή αναζήτησης. Παράλληλα, μόνο το 20% δοκιμάζει να αλλάξει το ερώτημα που έχει κάνει προκειμένου να βρει καλύτερα αποτελέσματα.

Στα κλασσικά συστήματα ανάκτησης πληροφορίας οι μετρικές που χρησιμοποιούνται για την αξιολόγηση είναι:

- Η ανάκληση
  - Το ποσοστό των σελίδων που έχουν επιστραφεί και είναι σχετικές
- Η ακρίβεια
  - Το ποσοστό των σχετικών σελίδων που έχουν επιστραφεί
- Η ακρίβεια στα πρώτα 10 αποτελέσματα

Σε ένα σύστημα όμως που έχει να κάνει με ανάκτηση πληροφορίας από το διαδίκτυο θα πρέπει:

Τα αποτελέσματα που επιστρέφονται να έχουν υψηλή σχετικότητα με το ερώτημα αλλά και υψηλή ποιότητα, δηλαδή με λίγα λόγια, θα πρέπει τα αποτελέσματα να είναι μόνο τα 'αναγκαία και απαραίτητα'.

Αυτό σημαίνει πως σε ένα τέτοιο σύστημα θα πρέπει να χρησιμοποιηθούν διαφορετικές μετρικές με τη βοήθεια των οποίων θα είναι σε θέση οι μηχανισμοί ανάκτησης πληροφορίας να μπορούν να αξιολογήσουν τα ερωτήματα των χρηστών και να επιστρέψουν τα πιο σωστά και αντιπροσωπευτικά αποτελέσματα.

Η αρχιτεκτονική των μηχανισμών ανάκτησης πληροφορίας από το Διαδίκτυο διαφέρει από την αρχιτεκτονική των μηχανισμών ανάκτησης πληροφορίας γενικά. Τα στοιχεία που είναι απαραίτητα σε ένα μηχανισμό ανάκτησης πληροφορίας είναι:

- Ο indexer
- Ο crawler και
- Ο query server.

Ο crawler χρησιμεύει στο να συλλέγονται σελίδες από το διαδίκτυο, ο indexer αναλαμβάνει να προβεί σε ανάλυση των ανακτημένων σελίδων και αναδόμηση αυτών προκειμένου να είναι εύκολη και εφικτή η αναζήτηση πάνω σε αυτές και τέλος ο query server είναι υπεύθυνος για την εξυπηρέτηση των ερωτημάτων από τους τελικούς χρήστες.

Αυτά τα τρία θεωρούνται τα βασικά δομικά στοιχεία ενός τέτοιου μηχανισμού ενώ δεν αποκλείεται σε σύνθετους μηχανισμούς ανάκτησης πληροφορίας από το διαδίκτυο να συναντήσουμε πολλά ακόμα υποσυστήματα αλλά και αναβαθμίσεις και αλλαγές στα συστήματα που ήδη περιγράψαμε. Αυτού του είδους τα συστήματα δημιουργούν ένα off-line αντίγραφο του διαδικτύου και εφαρμόζουν αλγορίθμους αναζήτησης στο αντίγραφο που διατηρούν. Άλλωστε είναι σχεδόν αδύνατη η δυναμική αναζήτηση στις δισεκατομμύρια σελίδες του διαδικτύου. Φυσικά τίθενται μία σειρά από προβλήματα τα οποία έχουν να κάνουν με το πόσο επικαιροποιημένο είναι το off-line αντίγραφο. Όσο πιο επικαιροποιημένο είναι τόσο ακριβέστερα αποτελέσματα θα εμφανίζονται. Ένα παράδειγμα που δείχνει την αδυναμία των μηχανισμών ανάκτησης πληροφορίας του διαδικτύου όπου παρουσιάζεται έντονα το φαινόμενο της μη επικαιροποιημένης πληροφορίας είναι οι πρώτες σελίδες των μεγάλων ειδησεογραφικών πρακτορείων. Οι σελίδες αυτές είναι κατασκευασμένες με τέτοιον τρόπο ώστε μπορεί μέσα σε 12 ώρες να έχει αλλάξει εντελώς το περιεχόμενο (κείμενο και εικόνες) στη συγκεκριμένη σελίδα. Προκειμένου ο μηχανισμός ανάκτησης πληροφορίας από το διαδίκτυο να είναι ενημερωμένος για τις συγκεκριμένες αλλαγές θα πρέπει να προσπελαύνει συνέχεια τη συγκεκριμένη σελίδα και να εντοπίζει αλλαγές, κάτι το οποίο είναι αδύνατο για τα σημερινά δεδομένα του χαώδους διαδικτύου.

Για την ακριβέστερη ανάκτηση πληροφορίας από το διαδίκτυο, η αδόμητη πληροφορία που ανακτάται από τις σελίδες που περιδιαβαίνει ο crawler θα πρέπει να δομηθεί με κατάλληλο τρόπο και να αποθηκεύεται σε τέτοια μορφή ώστε να μη χάσει τη συσχέτισή της από τα στοιχεία που την αποτελούν αλλά και από τις υπόλοιπες σελίδες που είναι όμοιές της. Τα στοιχεία που χρησιμοποιούνται για τη δόμηση των αποθηκευμένων σελίδων είναι συνήθως:

- Repository
  - Πρόκειται για το σημείο όπου αποθηκεύονται ολόκληρες οι σελίδες με τον HTML κώδικά τους.
- Document Index
  - Πρόκειται για πιο εξειδικευμένο χώρο αποθήκευσης πληροφορίας πια και όχι αρχείου όπου βέβαια υπάρχουν συσχετίσεις με τις σελίδες του repository καθώς και διάφορα στοιχεία checksum ή στατιστικά.
- Lexicon
  - Ένα λεξικό όπου είναι αποθηκευμένες περισσότερες από 20 εκατομμύρια λέξεις διαφόρων γλωσσών και χρησιμοποιούνται για ορθογραφικό έλεγχο των λέξεων των κειμένων.
- Hit Lists

  - Πρόκειται για λίστες που περιέχουν στοιχεία που αφορούν μονοπάτια που οδηγούν από μία σελίδα του διαδικτύου σε άλλη. Αυτές οι λίστες χρησιμοποιούνται σε συνδυασμό με εξειδικευμένους αλγορίθμους προκειμένου να προκύψουν συσχετίσεις και δεσμοί μεταξύ των σελίδων.

- Forward Index
  - Πρόκειται για λέξεις οι οποίες είναι ταξινομημένες βάσει ενός αύξοντα αριθμού που έχει ανατεθεί σε κάθε μία.

- Inverted Index
  - Είναι ακριβώς το ίδιο με το προηγούμενο μόνο που η ταξινόμηση γίνεται κατά φθίνουσα σειρά.

Οι περισσότεροι μηχανισμοί ανάκτησης πληροφορίας από το διαδίκτυο βασίζονται στον παραπάνω μηχανισμό που περιγράφηκε. Βασικός σκοπός τους είναι να λειτουργήσουν σαν μηχανές αναζήτησης και όχι για να προσφέρουν ένα ιστορικό του διαδικτύου. Επιπλέον, οι σελίδες που εμφανίζονται στον τελικό χρήστη δεν ταξινομούνται βάσει συσχέτισης με το ερώτημα αλλά βάσει ενός αριθμού που έχουν οι μηχανές αναζήτησης για κάθε σελίδα και ο οποίος δείχνει πόσο 'γνωστή' είναι η συγκεκριμένη σελίδα. Χαρακτηριστικό παράδειγμα αυτού είναι η μετρική page rank του Google. Έτσι αν μία σελίδα ενός προσωπικού δικτυακού τόπου για δελφίνια περιέχει τη λέξη 'δελφίνι' και την ίδια λέξη περιέχει κάποια σελίδα του CNN τότε οι μηχανές αναζήτησης στην αναζήτησή μας για τη λέξη δελφίνι θα βαθμολογήσουν περισσότερο τις σελίδες του πασίγνωστου CNN και λιγότερο τις σελίδες του προσωπικού δικτυακού τόπου.

2.2.5 Εξόρυξη γνώσης από αποθήκες δεδομένων

Η εξόρυξη γνώσης από μεγάλες *αποθήκες δεδομένων* που βρίσκονται στον παγκόσμιο ιστό, έχει εξελιχθεί σε ένα από τα βασικότερα ερευνητικά ζητήματα στον τομέα των βάσεων δεδομένων, των μηχανών γνώσης, της στατιστικής, καθώς επίσης και ως μία σημαντική ευκαιρία για καινοτομία στις επιχειρήσεις. Οι δικτυακές εφαρμογές που διαχειρίζονται μεγάλες αποθήκες δεδομένων, με σκοπό τη βελτίωση της ποιότητας των παρεχόμενων υπηρεσιών μέσω της μελέτης της συμπεριφοράς των πελατών και της εξαγωγής χρήσιμων συμπερασμάτων από αυτήν, αποτελούν αντικείμενο έρευνας.

Η τελευταία δεκαετία έχει επιφέρει μια αλματώδη αύξηση στην παραγωγή και συλλογή δεδομένων. Η πρόοδος στην τεχνολογία των βάσεων δεδομένων μας παρέχει νέες τεχνικές για την αποδοτική και αποτελεσματική συλλογή, αποθήκευση και διαχείριση των δεδομένων. Η δυνατότητα ανάλυσης και ερμηνείας των συνόλων δεδομένων και η εξαγωγή της 'χρήσιμης' γνώσης από αυτά έχει ξεπεράσει κάθε όριο, και η ανάγκη για μια νέα γενιά εργαλείων και τεχνικών για ευφυή ανάλυση των δεδομένων έχει δημιουργηθεί. Αυτή η ανάγκη έχει προσελκύσει την προσοχή των ερευνητών από διάφορες περιοχές (τεχνητή νοημοσύνη, στατιστική, αποθήκες δεδομένων, διαδραστική ανάλυση και επεξεργασία, έμπειρα συστήματα και οπτικοποίηση δεδομένων) και ένας νέος ερευνητικός τομέας δημιουργείται, γνωστός ως *εξόρυξη δεδομένων και γνώσης* (Data and Knowledge Mining).

2.2.6 Εξόρυξη γνώσης και δεδομένων

Η ανακάλυψη γνώσης από βάσεις δεδομένων, αναφέρεται στη διεργασία εξόρυξης γνώσης από τις μεγάλες αποθήκες δεδομένων οι οποίες συλλέγουν τα δεδομένα μέσα από την τεράστια κίνηση του παγκοσμίου ιστού. Ο όρος εξόρυξη δεδομένων χρησιμοποιείται ως συνώνυμο της ανακάλυψης γνώσης από βάσεις δεδομένων, καθώς επίσης και για αναφορά στις πραγματικές τεχνικές που χρησιμοποιούνται για την ανάλυση και την εξαγωγή της από διάφορα σύνολα δεδομένων. Πολλοί ερευνητές θεωρούν τον όρο εξόρυξη δεδομένων μη αντιπροσωπευτικό της διαδικασίας που περιγράφει, υποστηρίζοντας ότι ο όρος εξόρυξη γνώσης θα ήταν μια πιο κατάλληλη περιγραφή. Ο όρος *εξόρυξη δεδομένων* (*Data Mining*) είναι αυτός που έχει επικρατήσει και χαρακτηρίζει τη διαδικασία της εύρεσης δομών γνώσης οι οποίες περιγράφουν με ακρίβεια μεγάλα σύνολα πρωτογενών δεδομένων. Οι δομές αυτές αναδεικνύουν γνώση (συσχετίσεις ή κανόνες) που είναι κρυμμένοι μέσα στα δεδομένα και δεν μπορούν να εξαχθούν με 'γυμνό' μάτι. Οι προκύπτουσες δομές είναι πλούσιες σε σημασιολογία και εκμεταλλεύονται πιθανές κοινές ιδιότητες των πρωτογενών δεδομένων.

Οι δύο βασικοί στόχοι της εξόρυξης δεδομένων (γνώσης) είναι η εφαρμογή τεχνικών περιγραφής και πρόβλεψης σε μεγάλα σύνολα δεδομένων. Η πρόβλεψη στοχεύει στον υπολογισμό της μελλοντικής αξίας

ή στην πρόβλεψη της συμπεριφοράς κάποιων μεταβλητών που παρουσιάζουν ενδιαφέρον (π. χ. το ενδιαφέρον ενός αναγνώστη για διαφόρων κατηγοριών κείμενα) και οι οποίες βασίζονται στη συμπεριφορά άλλων μεταβλητών. Η περιγραφή επικεντρώνεται στην ανακάλυψη προτύπων και αναπαριστά τα δεδομένα μιας πολύπλοκης βάσης δεδομένων με έναν κατανοητό και αξιοποιήσιμο τρόπο. Ως προς την εξόρυξη γνώσης, η περιγραφή τείνει να είναι περισσότερο σημαντική από την πρόβλεψη.

2.2.7 Ανακάλυψη γνώσης από βάσεις δεδομένων σε σχέση με την εξόρυξη γνώσης και δεδομένων

Η ανακάλυψη γνώσης από βάσεις δεδομένων αναφέρεται σε ολόκληρη τη διαδικασία ανακάλυψης χρήσιμης πληροφορίας από μεγάλα σύνολα δεδομένων. Ένας τυπικός ορισμός δόθηκε από τους Frawley, Piatesky-Shaphiro & Matheus [64]:

Ορισμός 2.2.2. Ανακάλυψη γνώσης από βάσεις δεδομένων είναι η ντετερμινιστική διαδικασία αναγνώρισης έγκυρων, καινοτόμων, ενδεχομένως χρήσιμων και εν τέλει κατανοητών προτύπων στα δεδομένα.

Για την κατανόηση του παραπάνω ορισμού, παρατίθενται οι βασικές έννοιες των όρων πάνω στους οποίους είναι βασισμένος.

- Τα δεδομένα περιγράφουν οντότητες ή συσχετίσεις του πραγματικού κόσμου. Παραδείγματος χάριν θα μπορούσε να είναι ένα σύνολο ακατέργαστων κειμένων προερχόμενα από μια πηγή νέων του διαδικτύου.
- Ένα πρότυπο είναι μια έκφραση E σε μια γλώσσα L η οποία περιγράφει ένα υποσύνολο δεδομένων $F_E \subseteq F$ εκμεταλλευόμενο κοινές ιδιότητες των δεδομένων του.
- Η διαδικασία ανακάλυψη γνώσης από βάσεις δεδομένων είναι μια διαδικασία πολλαπλών βημάτων, η οποία περιλαμβάνει την προ-επεξεργασία των δεδομένων, την αναζήτηση των προτύπων και την αξιολόγηση της εξαγόμενης γνώσης.
- Εγκυρότητα. Το εξαγόμενο πρότυπο (π. χ. περίληψη κειμένου) θα πρέπει να είναι συνεπές σε νέα δεδομένα με κάποιο βαθμό βεβαιότητας. Το ζήτημα της εγκυρότητας αποτελεί ένα από τα βασικά προβλήματα και αντικείμενο έρευνας στην εξόρυξη δεδομένων / πληροφορίας.
- Πιθανά χρήσιμο. Η εξαγωγή των προτύπων θα πρέπει να ακολουθείται από μερικές χρήσιμες διεργασίες όπως η αξιολόγησή τους από κάποιες συναρτήσεις χρησιμότητας. Για παράδειγμα η αυτόματη περίληψη ενός κειμένου θα πρέπει να μπορεί να αξιολογηθεί ως προς την χρησιμότητα / σαφήνειά και την πιστότητά του όσον αφορά το νόημα σε σχέση με το αρχικό κείμενο. Επίσης, θα ήταν χρήσιμο να εμπλουτιστεί η σημασιολογία των προτύπων, διατηρώντας όσο το δυνατόν περισσότερη γνώση από τα αρχικά δεδομένα η οποία μπορεί να φανεί χρήσιμη για τη λήψη αποφάσεων.
- Τελικά κατανοητό. Ο στόχος της εξόρυξης γνώσης είναι να προσδιοριστούν τα πρότυπα και να γίνουν κατανοητά, ώστε να μπορούν να οδηγήσουν ακόμη και τους μη ειδικούς σε χρήσιμα συμπεράσματα και αποφάσεις.

Η διαδικασία ανακάλυψη γνώσης είναι μια διαλογική και επαναληπτική διαδικασία που αποτελείται από μια σειρά από τα ακόλουθα βήματα:

- Την ανάπτυξη και κατανόηση της περιοχής της εφαρμογής, της σχετικά προγενέστερης γνώσης του προς εξέταση τομέα και τους στόχους του τελικού χρήστη.
- Την ολοκλήρωση των δεδομένων. Υπάρχουν διαφορετικά είδη αποθηκών πληροφοριών που μπορούν να χρησιμοποιηθούν στη διαδικασία εξόρυξης γνώσης. Κατά συνέπεια οι πολλαπλές πηγές δεδομένων μπορούν να συνδυαστούν καθορίζοντας το σύνολο στο οποίο τελικά η διαδικασία εξόρυξης πρόκειται να εφαρμοστεί.
- Τη δημιουργία του στόχου-συνόλου δεδομένων. Επιλογή του συνόλου δεδομένων (δηλαδή μεταβλητές, δείγματα δεδομένων) στο οποίο η διαδικασία εξόρυξης πρόκειται να εκτελεστεί.

- Τον καθαρισμό και την προ-επεξεργασία δεδομένων. Αυτό το βήμα περιλαμβάνει βασικές διαδικασίες όπως η αφαίρεση του θορύβου, η συλλογή των απαραίτητων πληροφοριών για τη διαμόρφωση ή τη μέτρηση του θορύβου, η απόφαση σχετικά με τις στρατηγικές διαχείρισης των ελλειπόντων πεδίων δεδομένων.

- Τον μετασχηματισμό των δεδομένων. Τα δεδομένα μετασχηματίζονται ή παγιώνονται σε μορφές κατάλληλες για εξόρυξη. Χρήση των μεθόδων μείωσης διαστάσεων ή μετασχηματισμού για τη μείωση του αριθμού των υπό εξέταση μεταβλητών ή την εύρεση κατάλληλης αντιπροσώπευσης των δεδομένων χωρίς μεταβλητές.

- Την επιλογή των στόχων και των αλγορίθμων εξόρυξης δεδομένων. Σε αυτό το βήμα αποφασίζουμε το στόχο της διαδικασίας εξόρυξης γνώσης, επιλέγοντας τους στόχους εξόρυξης δεδομένων που θέλουμε να επιτύχουμε. Επίσης, επιλέγονται οι μέθοδοι που θα χρησιμοποιηθούν. Αυτό περιλαμβάνει την επιλογή του κατάλληλου μοντέλου και παραμέτρων.

- Την εξόρυξη δεδομένων. Εφαρμόζοντας ευφυείς μεθόδους, ψάχνουμε για ενδιαφέροντα πρότυπα γνώσης. Τα πρότυπα θα μπορούσαν να είναι μιας συγκεκριμένης αντιπροσωπευτικής μορφής ή ενός συνόλου τέτοιων αντιπροσωπεύσεων, όπως κανόνες κατηγοριοποίησης, δέντρα, συσταδοποίηση, κλπ Η απόδοση και τα αποτελέσματα της μεθόδου εξόρυξης δεδομένων εξαρτώνται από τα προηγούμενα βήματα.

- Την αξιολόγηση των προτύπων. Τα εξαγόμενα πρότυπα αξιολογούνται με κάποια μέτρα, προκειμένου να προσδιοριστούν τα πρότυπα τα οποία αντιπροσωπεύουν τη γνώση, δηλαδή τα αληθινά ενδιαφέροντα πρότυπα.

- Τη σταθεροποίηση και παρουσίαση της γνώσης. Σε αυτό το βήμα, η εξορυγμένη γνώση ενσωματώνεται το σύστημα ή απλά την απεικόνισή μας και κάποιες τεχνικές αντιπροσώπευσης γνώσης χρησιμοποιούνται για να παρουσιάσουν την εξορυγμένη γνώση στο χρήστη. Επίσης, ελέγχουμε για επίλυση τυχών συγκρούσεων με προηγούμενη εξορυγμένη γνώση.

Η εξόρυξη δεδομένων ως βήμα της διαδικασίας εξόρυξης γνώσης ενδιαφέρεται κυρίως για τις μεθοδολογίες και τις τεχνικές εξαγωγής προτύπων δεδομένων ή τις περιγραφές δεδομένων από τις μεγάλες αποθήκες δεδομένων. Αφ' ετέρου, η διαδικασία εξόρυξης γνώσης περιλαμβάνει την αξιολόγηση και την ερμηνεία των προτύπων. Επίσης περιλαμβάνει την επιλογή της κωδικοποίησης των προτύπων, της προ-επεξεργασίας, της δειγματοληψίας και του μετασχηματισμού των δεδομένων πριν από το βήμα της εξόρυξης των δεδομένων.

2.2.8 Η διαδικασία εξόρυξης δεδομένων

Η *εξόρυξη δεδομένων* περιλαμβάνει τα μοντέλα συναρμολογήσεων των υπό εξέταση δεδομένων, ή εναλλακτικά την εξαγωγή των προτύπων από αυτά. Ουσιαστικά, οι παράμετροι του μοντέλου είναι γνωστές από τα δεδομένα ή τα πρότυπα που προσδιορίζονται, αντιπροσωπεύουν τη γνώση που έχει εξαχθεί από ένα σύνολο δεδομένων.

Υπάρχει μια μεγάλη συλλογή αλγορίθμων εξόρυξης δεδομένων, πολλοί από τους οποίους χρησιμοποιούν έννοιες και τεχνικές από διαφορετικούς τομείς όπως η στατιστική, η αναγνώριση προτύπων, η μηχανική μάθηση, οι αλγόριθμοι και οι βάσεις δεδομένων. Μια θεμελιώδης ιδιότητα των αλγορίθμων εξόρυξης δεδομένων, και αυτή που διαφοροποιεί τους περισσότερους από αυτούς από άλλες παρόμοιες τεχνικές που υιοθετούνται στη μηχανική μάθηση και τη στατιστική, είναι ότι οι αλγόριθμοι εξόρυξης δεδομένων έχουν σχεδιαστεί με έμφαση στην εξελιξιμότητα όσον αφορά το μέγεθος του συνόλου δεδομένων εισαγωγής. Η πλειοψηφία των αλγορίθμων εξόρυξης δεδομένων θα μπορούσε να περιγραφεί σε υψηλό επίπεδο με τον όρο ενός απλού πλαισίου. Συγκεκριμένα μπορούν να αντιμετωπισθούν ως σύνθεση των τριών ακολούθων συστατικών:

- Την περιγραφή του μοντέλου. Υπάρχουν δύο παράγοντες σχετικοί με το μοντέλο:

  – Η λειτουργία του μοντέλου. Καθορίζει τους βασικούς στόχους κατά τη διάρκεια της διαδικασίας εξόρυξης δεδομένων.

  - Η παραστατική μορφή του μοντέλου. Η απεικόνιση του μοντέλου καθορίζει και το ταίριασμά του με την απεικόνιση των δεδομένων και τη δυνατότητα να ερμηνευθεί το μοντέλο με κατανοητούς όρους. Χαρακτηριστικά, πιο περίπλοκα μοντέλα ταιριάζουν καλύτερα στα δεδομένα αλλά μπορεί να είναι δυσκολότερο να γίνουν κατανοητά και να ανταποκριθούν σε πραγματικές συνθήκες.

- Την αξιολόγηση του μοντέλου. Με βάση κάποια κριτήρια αξιολόγησης (π. χ. μέγιστη πιθανότητα) θα μπορούσαμε να καθορίσουμε πόσο καλά ένα συγκεκριμένο μοντέλο ταιριάζει με τα κριτήρια της διαδικασίας εξόρυξης γνώσης. Γενικά, η αξιολόγηση του μοντέλου αναφέρεται και στην εγκυρότητα των προτύπων και στην αξιολόγηση της ακρίβειας, της χρησιμότητας και της δυνατότητας κατανόησης του μοντέλου.

- Τους αλγορίθμους αναζήτησης. Αναφέρεται στην προδιαγραφή ενός αλγορίθμου να βρίσκει συγκεκριμένα μοντέλα και παραμέτρους, δοσμένου ενός συνόλου δεδομένων, μιας οικογένειας μοντέλων και ενός κριτηρίου αξιολόγησης. Υπάρχουν δύο τύποι αλγορίθμων αναζήτησης:

  - Αυτοί που αναζητούν παραμέτρους. Αυτός ο τύπος αλγορίθμων ψάχνει για παραμέτρους, οι οποίες βελτιστοποιούν ένα κριτήριο αξιολόγησης για το μοντέλο. Οι αλγόριθμοι εκτελούν το στόχο αναζήτησης παίρνοντας ως είσοδο ένα σύνολο δεδομένων και μια απεικόνιση μοντέλου.
  - Αυτοί που αναζητούν μοντέλα. Εκτελούν μια επαναληπτική διαδικασία αναζήτησης για την αντιπροσώπευση των δεδομένων. Για κάποια συγκεκριμένη απεικόνιση του μοντέλου, εφαρμόζεται η μέθοδος αναζήτησης παραμέτρων και η ποιότητα των αποτελεσμάτων αξιολογείται.

2.2.9 Κατηγορίες μεθόδων εξόρυξης πληροφορίας

Τα τελευταία χρόνια διάφορες τεχνικές και μέθοδοι εξόρυξης δεδομένων έχουν αναπτυχθεί. Διαφορετικά κριτήρια κατηγοριοποίησης μπορούν να χρησιμοποιηθούν για να κατηγοριοποιήσουν τις μεθόδους και τα συστήματα εξόρυξης δεδομένων, βασισμένες στους τύπους των βάσεων δεδομένων που θα χρησιμοποιηθούν, τους τύπους γνώσης που θα εξαχθούν και τις τεχνικές που θα εφαρμοστούν. Η κατηγοριοποίηση των μεθόδων εξόρυξης πληροφορίας βασίζεται στα ακόλουθα κριτήρια:

- Είδος πηγής δεδομένων που χρησιμοποιείται. Π. χ. ένα σύστημα εξόρυξης πληροφορίας που χρησιμοποιεί δεδομένα μια σχεσιακής βάσης δεδομένων μπορεί να ονομαστεί σχεσιακό.

- Είδος γνώσης που εξάγεται. Από ένα σύστημα εξόρυξης δεδομένων θα μπορούσαν να εξαχθούν διάφορα είδη γνώσης, όπως κανόνες συσχέτισης, συσταδοποίηση, κανόνες κατηγοριοποίησης, χαρακτηριστικοί κανόνες. Ένα σύστημα εξόρυξης δεδομένων θα μπορούσε να ταξινομηθεί σύμφωνα με το επίπεδο γενίκευσης της εξαγόμενης γνώσης, η οποία θα μπορούσε να είναι γενική, πρώτου επιπέδου ή πολυεπίπεδη γνώση.

- Είδος χρησιμοποιούμενων τεχνικών. Τα συστήματα εξόρυξης δεδομένων θα μπορούσαν να ταξινομηθούν σύμφωνα με τις χρησιμοποιούμενες τεχνικές εξόρυξης δεδομένων. Για παράδειγμα, θα μπορούσαν να ταξινομηθούν σε αυτόνομα συστήματα, συστήματα προσανατολισμένα στα δεδομένα, συστήματα οδηγούμενα από ερωταποκρίσεις καθώς και διαλογικά συστήματα. Επίσης, σύμφωνα με την προσέγγιση που χρησιμοποιείται θα μπορούσαν να ταξινομηθούν σε συστήματα γενικής εξόρυξης, εξόρυξης βασισμένης στα πρότυπα, εξόρυξης βασισμένης στη στατιστική ή στα μαθηματικά κλπ.

2.2.10 Εύρεση προτύπων συσχέτισης

Η ανακάλυψη χρήσιμης πληροφορίας, μέσα σε συγκεκριμένα έγγραφα, αποτελεί το πεδίο δράσης της διαδικασίας της εύρεσης *προτύπων συσχέτισης (Association Patterns)* . Οι Arimura Hiroki, Wataki Atsushi, Fujino Ryoichi και Arikawa Setsuo [39], μελέτησαν την ανακάλυψη πολύ απλών προτύπων, που τα ονόμασαν πρότυπα συσχέτισης ζευγών λέξεων k-εγγύτητας (k-proximity two-words association patterns). Σε μία δεδομένη συλλογή κειμένων και με τη χρήση μιας αντικειμενικής συνθήκης, ορίζεται το πρότυπο συσχέτισης. Το πρότυπο αυτό, εκφράζει ένα κανόνα που αναφέρει ότι αν βρεθεί η υπολέξη που περιέχεται στο πρότυπο, ακολουθούμενη από μία άλλη δεδομένη υπολέξη, σε συγκεκριμένη απόσταση γραμμάτων, τότε η αντικειμενική συνθήκη θα διατηρηθεί με μεγάλη πιθανότητα.

Οι κανόνες αυτοί είναι πολύ ευέλικτοι για την περιγραφή των τοπικών ομοιοτήτων που περιέχονται στα δεδομένα του κειμένου. Το είδος των κανόνων αυτών, χρησιμοποιείται για παράδειγμα στην βιοπληροφορική, στην βιβλιογραφική έρευνα και στην έρευνα στο διαδίκτυο. Ως γενικό πλαίσιο εργασίας, ο αλγόριθμος ανακάλυψης προτύπων λαμβάνει ένα σύνολο δειγμάτων με μία συγκεκριμένη συνθήκη και βρίσκει όλα ή μερικά από τα πρότυπα, τα οποία μεγιστοποιούν ένα συγκεκριμένο κριτήριο.

Διακρίνουμε το πρόβλημα του προτύπου βέλτιστης εμπιστοσύνης όπου, δεδομένου ενός συνόλου από έγγραφα και με μία αντικειμενική συνθήκη για αυτό το σύνολο, υπολογίζεται το πρότυπο που μεγιστοποιεί την τιμή των κριτηρίων που έχουν τεθεί για τα συγκεκριμένα έγγραφα. Ένα δεύτερο πρόβλημα, αναφέρεται στην ελαχιστοποίηση του εμπειρικού λάθους, όπου αναζητείται ένα πρότυπο που θα ελαχιστοποιεί τον αριθμό των εγγράφων που έχουν επεξεργαστεί με λάθος τρόπο.

Χαρακτηριστικές εφαρμογές που χρησιμοποιούν την εύρεση προτύπων συσχέτισης, είναι αυτές που αναλύουν απλά έγγραφα κειμένου, όπως προτείνουν και οι Montes-y-Gomez M., Gelbukh A. και Lopez-Lopez A. [100]. Προσπαθούν να ανακαλύψουν τις σχέσεις που υπάρχουν ανάμεσα στα διάφορα θέματα που παρουσιάζονται σε αφημερίδες. Επιχειρούν να ανακαλύψουν τον τρόπο που τα θέματα της λεγόμενης πρώτης σελίδας, επηρεάζουν και όλα τα υπόλοιπα θέματα της ειδησιογραφίας. Οι συσχετίσεις που υπάρχουν ανάμεσα στα διάφορα ειδησιογραφικά θέματα, καλούνται εφήμερες (Ephemeral Associations). Άλλη χαρακτηριστική εφαρμογή, αποτελεί η ανακάλυψη προτύπων σε σύνολα ακολουθιών DNA, που προτείνουν οι Kiem Hoang και Phuc Do [73]. Μελετούν υποακολουθείες που εμφανίζονται πολύ συχνά στο σύνολο των ακολουθιών DNA, για την ανακάλυψη εκείνων των κανόνων συσχέτισης, που βασίζονται στην επανάληψη.

2.2.11 Ανάκτηση γνώσης από βάσεις δεδομένων

Η *ανάκτηση γνώσης από βάσεις δεδομένων* (Knowledge Discovery in Databases - KDD) είναι η μη τετριμμένη διαδικασία της αναγνώρισης έγκυρων, καινότυπων, ενδεχόμενα χρήσιμων και τελικά κατανοητών προτύπων δεδομένων. Τα ακατέργαστα δεδομένα είναι πάντοτε 'ακάθαρτα' με την έννοια ότι πάντα θα υπάρχουν διπλοεγγραφές, ελλειπή πεδία και μη ακριβές τιμές δεδομένων. Είναι επιθυμητό επομένως, τα αποτελέσματα των αναζητήσεων να πρέπει να περάσουν από κάποιο στάδιο εκκαθάρισης πριν παρουσιαστούν στον χρήστη. Η εκκαθάριση δεδομένων στην KDD διαδικασία είναι ένα βασικό βήμα για την αφαίρεση του θορύβου και των outliers[1], την συγκέντρωση των σχετικών πληροφοριών για μοντελοποίηση του θορύβου και την λήψη αποφάσεων για τα ελλειπή δεδομένα.

Τα καθαρά δεδομένα υπονοούν και σχετικά δεδομένα παρότι η σχετικότητα των δεδομένων είναι συνήθως υποκειμενική. Είναι όμως γεγονός ότι μια ακριβής περίληψη ενός κειμένου μπορεί να χρησιμοποιηθεί για να εκτιμηθεί η σχετικότητα ή μη του αρχικού κειμένου με τα ενδιαφέροντα του χρήστη. Παράλληλα, μια προηγούμενη αντιστοίχηση των εξαγομένων κειμένων με ορισμένα πεδία ενδιαφέροντος μπορεί να βοηθήσει στον εντοπισμό των outliers. Αυτό σημαίνει ότι εκείνα τα έγγραφα που δεν εμπίπτουν στις κατηγορίες ενδιαφέροντος του χρήστη, μπορούν να αγνοηθούν.

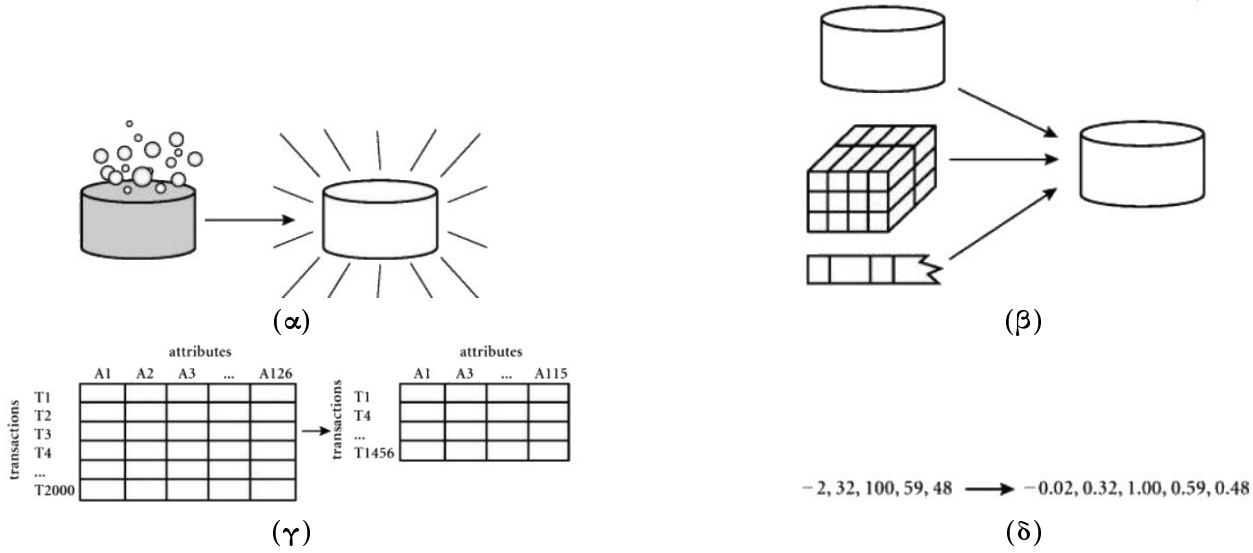
2.3 Προεπεξεργασία Δεδομένων

Τα δεδομένα που κατακλύζουν τις σύγχρονες βάσεις δεδομένων και τον παγκόσμιο ιστό σήμερα, είναι πολύ επιρρεπή σε θόρυβο, σε ανεπάρκεια ή συνοχή λόγω κυρίως του τεράστιου όγκου και της ετερογένειας τον πηγών τους. Δεδομένα χαμηλής ποιότητας οδηγούν σε χαμηλής ποιότητας εξόρυξη πληροφορίας. Το θεμελιώδες ερώτημα που τίθεται είναι: πώς μπορούν να προεπεξεργαστούν τα δεδομένα, ώστε να βελτιωθεί η ποιότητά τους και επομένως τα αποτελέσματα της εξόρυξης πληροφορίας;

Υπάρχει ένα πλήθος μεθόδων που χρησιμοποιούνται για την προεπεξεργασία δεδομένων (Σχήμα 2.3). Το καθάρισμα δεδομένων μπορεί να έχει εφαρμογή στην αφαίρεση του θορύβου από τα δεδομένα και στην διόρθωση των ασυνεπειών σε αυτά. Η ολοκλήρωση των δεδομένων συνενώνει δεδομένα από διάφορες πηγές σε συναφή αποθήκη δεδομένων, όπως π. χ. μια βάση δεδομένων. Ο μετασχηματισμός των δεδομένων, όπως η κανονικοποίηση μπορεί να χρησιμοποιηθεί από τη διαδικασία προεπεξεργασίας δεδομένων. Για παράδειγμα, η κανονικοποίηση μπορεί να βελτιώσει την ακρίβεια και την αποτελεσματικότητα των αλγορίθμων εξόρυξης δεδομένων ενσωματώνοντας μετρικές απόστασης. Η αφαίρεση δεδομένων, μπορεί να μειώσει το μέγεθος

[1] δεδομένα που βρίσκονται εκτός του διαστήματος τυπικής απόκλισης των υπολοίπων δεδομένων και ως εκ' τούτου αποτυγχάνουν να αναπαραστήσουν σωστά την πληροφορία

των δεδομένων, συναθροίζοντας, απαλείφοντας τα πλεονάζοντα χαρακτηριστικά, ή ομαδοποιώντας τα δεδομένα. Αυτές οι τεχνικές δεν είναι αμοιβαία αποκλειόμενες· μπορούν να δουλέψουν μαζί. Για παράδειγμα, το καθάρισμα δεδομένων μπορεί να περιλαμβάνει μετασχηματισμούς για την διόρθωση λανθασμένων δεδομένων. Οι τεχνικές προεπεξεργασίας δεδομένων, όταν εφαρμόζονται πριν την εξόρυξη πληροφορίας, μπορούν να βελτιώσουν σημαντικά την ποιότητα της πληροφορίας που εξορύσσεται ή τον χρόνο που απαιτείται γι' αυτή τη διαδικασία.



Σχήμα 2.3: Τεχνικές προεπεξεργασίας δεδομένων (α)Καθαρισμός δεδομένων (β)Ολοκλήρωση δεδομένων (γ)Αφαίρεση δεδομένων (δ)Μετασχηματισμός δεδομένων

2.3.1 Αφαίρεση σημείων στίξης

Τα *σημεία στίξης* (*punctuation*) ενός κειμένου δεν προσδίδουν σημασιολογική πληροφορία σε αυτό και άρα δεν δεικτοδοτούνται. Είναι επομένως αναγκαίο, ένα σύστημα ανάκτησης πληροφορίας να αφαιρεί κάθε σημείο στίξης από το αρχικό κείμενο σε πρώιμα στάδια της προεπεξεργασίας. Ιδιαίτερη μέριμνα πρέπει να λαμβάνεται ώστε να συγκρατείται το τέλος της κάθε πρότασης (π. χ. με κάποιο άλλο διαχωριστικό πέραν της τελείας) ώστε να είναι δυνατός ο μετέπειτα διαχωρισμός των προτάσεων. Η διαδικασία θα πρέπει να λαμβάνει όσο το δυνατός καλύτερα υπ' όψιν τις γλωσσολογικές ιδιομορφίες της εκάστοτε γλώσσας ώστε να μην προκύπτουν λάθη κατά τη διαδικασία της αφαίρεσης των σημείων στίξης. Ορισμένα παραδείγματα:

- Ne'er: χρήση language-specific πηγών για τον κατάλληλο μετασχηματισμό
- State-of-the-art: διαχωρισμός λέξεων με παύλες σε ξεχωριστά tokens
- U.S.A. vs. USA: απομάκρυνση ενδιάμεσων τελειών σε ακρωνύμια

2.3.2 Αφαίρεση αριθμών

Γενικά, οι αριθμοί ενός κειμένου δεν δεικτοδοτούνται (τουλάχιστον όχι όπως το υπόλοιπο κείμενο) για λόγους παρόμοιους με αυτών των σημείων στίξης. Η αντιμετώπισή τους μπορεί να ποικίλει από IR σε IR σύστημα και εξαρτάται κυρίως από τις απαιτήσεις που θέτονται. Σπάνια χρειάζεται να ανακτηθεί μια ημερομηνία π. χ. από ένα μεγάλο κείμενο αλλά η πληροφορία αυτή μπορεί να αποθηκευθεί ως meta-δεδομένο για το κείμενο.

2.3.3 Κεφαλαία γράμματα

Η διάκριση μεταξύ κεφαλαίων και μικρών γραμμάτων, αμελητέα μόνο σημασιολογική πληροφορία μπορεί να δώσει για το κείμενο. Για το λόγο αυτό, και για ομοιομορφία των προς επεξεργασία λέξεων, όλα τα κεφαλαία γράμματα συνήθως μετασχηματίζονται σε μικρά.

2.3.4 *Stopwords*

Τα *Stopwords* είναι λέξεις οι οποίες περιέχουν μικρής σημασίας πληροφορία για το κείμενο. Είναι ως επί το πλείστον 'λειτουργικές' λέξεις οι οποίες εμφανίζονται σε ένα μεγάλο μέρος των κειμένων και ως εκ' τούτου, περιέχουν μικρή ικανότητα διάκρισης για δήλωση συσχέτισης. Στην διαδικασία της ανάκτησης πληροφορίας, τα Stopwords συνήθως αγνοούνται και για λόγους αποδοτικότητας, αφού η αποθήκευση των Stopwords σε ένα ευρετήριο λαμβάνει σημαντικό χώρο λόγω της υψηλής συχνότητας εμφάνισής τους. Η αφαίρεση των Stopwords από ένα κείμενο θα μπορούσαμε να πούμε ότι είναι μια τεχνική αφαίρεσης δεδομένων η οποία απαλλάσσει το κείμενο από ένα σημαντικό μέγεθος μη-χρήσιμης πληροφορίας.

2.3.5 *Stemming*

Η διαδικασία του *Stemming* εξάγει τη μορφολογική ρίζα κάθε λέξης. Παράλληλα, και ανάλογα με τη λίστα κανόνων που χρησιμοποιούνται, η διαδικασία του Stemming μπορεί να περιλαμβάνει και τη λημματοποίηση του κειμένου· την εύρεση δηλαδή του λήμματος κλιτών λέξεων (π. χ. children→child). Σε καθολικές μηχανές αναζήτησης, το βασικό πρόβλημα της διαδικασίας του Stemming είναι ότι είναι γλώσσο-εξαρτώμενη, και ενώ για την αγγλική γλώσσα υπάρχουν *Stemmers* βασισμένοι σε κανόνες, για άλλες γλώσσες είναι δύσκολη η ανάπτυξη.

2.4 Περίληψη Πληροφορίας

Η διαδικασία της περίληψης κειμένου *(Text Summarization)* , αποσκοπεί στην παρουσίαση των κύριων σημείων ενός εγγράφου, σε μία περιεκτική μορφή. Μία πραγματική περίληψη, θα πρέπει να εκφράζει την ουσία του εγγράφου, αποκαλύπτοντας το βαθύτερο νόημα του περιεχομένου του. Σκοπός της είναι, η ανακάλυψη ενδιαφέρουσας και απροσδόκητης πληροφορίας. Σύμφωνα με τον Crangle Colleen [54], υπάρχουν δύο κύριες αντιλήψεις για την εξαγόμενη περίληψη του αρχικού κειμένου. Η πρώτη αναφέρει ότι η περίληψη θα περιέχει προτάσεις οι οποίες περιέχονται μόνο στο αρχικό κείμενο. Η δεύτερη είναι πιο σύνθετη και αναφέρει ότι εκτός των αρχικών προτάσεων του κειμένου, είναι δυνατόν να υπάρχουν και άλλες, κατασκευασμένες από τον μηχανισμό περίληψης. Οι προτάσεις αυτές, είτε θα δημιουργούνται με τη χρήση τμημάτων των αρχικών προτάσεων, είτε με την επεξεργασία των αρχικών και την παραγωγή νέων, που δεν θα περιέχουν τμήματα, που υπάρχουν στις αρχικές προτάσεις. Μπορούμε να αναφερθούμε στις δύο αυτές διαφορετικές κλάσεις τεχνικών περίληψης κειμένου χρησιμοποιώντας τις έννοιες αφαίρεση και εξαγωγή.

Σε αντίθεση με τις τεχνικές της αφαίρεσης, οι οποίες απαιτούν τεχνικές *Natural Language Processing - NLP*, συμπεριλαμβανομένων γραμματικών και λεξικών για την ανάλυση του κειμένου, η εξαγωγή μπορεί να θεωρηθεί ως μια διεργασία επιλογής σημαντικών αποσπασμάτων (προτάσεων, παραγράφων, κ.λπ.) από το αρχικό κείμενο και συνένωσής του σε μια νέα πιο σύντομη έκδοση.

Οι περιλήψεις κειμένων μπορεί να είναι είτε συσχετιζόμενες με κάποιο ερώτημα χρήστη (προτιμήσεις του χρήστη), είτε γενικές. Το πρώτο είδος επιστρέφει περιεχόμενο του κειμένου που ανταποκρίνεται στις προτιμήσεις του χρήστη, μια διαδικασία που περιέχει πολλά κοινά με την διαδικασία ανάκτησης κειμένων και ως εκ' τούτου, οι αλγόριθμοι που χρησιμοποιούνται συνήθως πηγάζουν από αυτή. Από την άλλη μεριά, μια γενική περίληψη παρέχει μια συνολική άποψη για τα περιεχόμενα του κειμένου. Μια καλή γενική περίληψη πρέπει να περιέχει τα βασικά σημεία του κειμένου διατηρώντας παράλληλα τον πλεονασμό στο ελάχιστο. Σε αυτή την εργασία αξιοποιούνται τεχνικές που αφορούν και τα δύο είδη περίληψης: α)γενική και β)προσωποποιημένη στο χρήστη.

2.4.1 Χρησιμότητα της περίληψης κειμένου

Στο επίκαιρο σενάριο της συνδυαστικής έκρηξης της πληροφορίας που εμφανίζεται στις μέρες μας, η αναζήτηση για καλύτερες τεχνικές εξαγωγής πληροφορίας (Information Retrieval - IR) συνεχίζει να γοητεύει τους επιστήμονες της πληροφορικής. Παρότι όμως τα σύγχρονα συστήματα για αναζήτηση και ανάκτηση πληροφορίας είναι ικανά να ανακτούν χιλιάδες εγγράφων στην επιφάνεια εργασίας των χρηστών και μάλιστα σε πολύ σύντομο χρονικό διάστημα, απέχουν πολύ από την ιδανική λύση. Ο χρήστης πρέπει να κάνει πολλές κρίσεις που έχουν να κάνουν με τη σχετικότητα των εγγράφων με τα ενδιαφέροντά του 'ξαφρίζοντας' μέσα από πολλαπλά έγγραφα, τα περισσότερα εκ' των οποίων είναι άσχετα. Η διαδικασία αυτή είναι ιδιαίτερα επίπονη και χρονοβόρα για τον χρήστη που επιθυμεί να εντοπίσει γρήγορα και εύκολα το κείμενο που επιθυμεί.

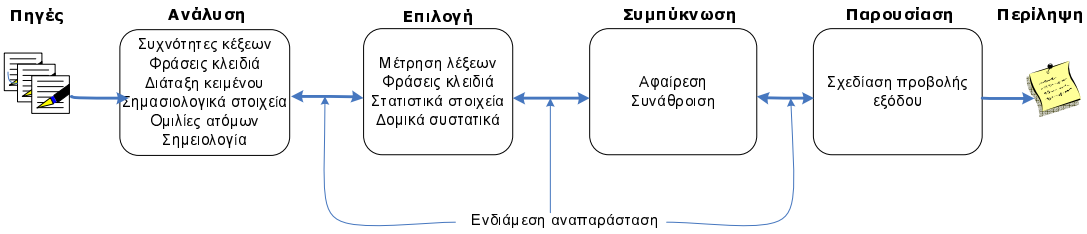
Είναι λοιπόν προφανές ότι η κοινότητα των χρηστών θα ωφεληθεί σημαντικά εάν τα ανακτημένα έγγραφα 'συμπυκνωθούν' με κάποιον τρόπο και παρουσιαστούν πίσω στον τελικό χρήστη με τη μορφή αναγνώσιμης και εύκολα διαχειρίσιμης περίληψης. Δυστυχώς, οι απαιτήσεις για ακρίβεια και ανάκληση επιβάλουν αντικρουόμενες απαιτήσεις στο σύστημα. Σε αυτό το ζήτημα είναι εύλογο να θεωρηθεί ότι μια αναζήτηση με υψηλή ακρίβεια με τα ενδιαφέροντα του χρήστη είναι πιο πιθανό να ικανοποιήσει τον μέσο χρήστη σε σχέση με μια εξαντλητική αναζήτηση ενός μεγάλου πλήθους κειμένων. Αυτά τα θέματα, μαζί με την αυξανόμενη ποικιλία των συλλογών κειμένων, αναδεικνύουν τον τομέα της αυτοματοποιημένης περίληψης κειμένων ως έναν από τους βασικότερους της ανάκτησης πληροφορίας.

Οι περιλήψεις κειμένων, μπορούν να χρησιμοποιηθούν από αναλυτές πληροφοριών, έτσι ώστε να είναι σε θέση να γνωρίζουν αν θα πρέπει να μελετήσουν κάποια κείμενα στο σύνολο τους, και κάποια άλλα με διαφορετικό και πιο περιεκτικό τρόπο. Οι περιλήψεις μπορούν να αποκαλύψουν ομοιότητες στο περιεχόμενο των κειμένων, οι οποίες μπορούν να χρησιμοποιηθούν για την μετέπειτα ομαδοποίηση ή κατηγοριοποίηση των εγγράφων. Η διαδικασία της κατηγοριοποίησης ή ομαδοποίησης των περιλήψεων περισσοτέρων του ενός εγγράφου, μέσα σε μία συλλογή, μπορεί να αποκαλύψει αναπάντεχες σχέσεις μεταξύ των εγγράφων. Επιπλέον, η περίληψη μιας συλλογής από σχετιζόμενα έγγραφα, που έχουν επεξεργαστεί μαζί, μπορεί να αποκαλύψει αθροιστική πληροφορία, που υπάρχει μόνο στο επίπεδο της συλλογής των εγγράφων.

2.4.2 Η διαδικασία της περίληψης

Μια αποτελεσματική περίληψη κειμένου εντοπίζει την σημαντική πληροφορία από μια ή περισσότερες πηγές και παράγει μια συντομευμένη έκδοση της αρχικής πληροφορίας. Η διαδικασία της αυτοματοποιημένης περίληψης περικλείει τουλάχιστον τέσσερα διακριτά στάδια επεξεργασίας (Εικόνα 2.4):

1. Ανάλυση του κειμένου
2. Αναγνώριση / Εντοπισμός των σημαντικών τμημάτων του κειμένου
3. Συμπύκνωση πληροφορίας και
4. Παραγωγή της αναπαράστασης της περίληψης που προκύπτει.



Σχήμα 2.4: Γενική διαδικασία παραγωγής περίληψης.

2.4.3 Αξιολόγηση της εξαγόμενης περίληψης

Η *αξιολόγηση της περίληψης* που προκύπτει από ένα σύστημα αυτόματης εξαγωγής περίληψης, είναι μια εργασία εξίσου σημαντική με την ίδια τη διαδικασία εξαγωγής. Η αξιολόγηση όμως πρέπει να είναι 'φθηνή', από άποψη υπολογιστικού κόστους και συνάμα εφαρμόσιμη και αποτελεσματική για ένα ευρύ φάσμα κειμένων που εισέρχονται στο σύστημα. Στη συνέχεια περιγράφονται οι πλέον συνηθισμένοι τρόποι αξιολόγησης μιας περίληψης.

Αξιολόγηση με συσχέτιση προτάσεων

Η συσχέτιση των εξαγόμενων προτάσεων με το αρχικό κείμενο περιλαμβάνει μετρικές ακρίβειας και ανάκλησης. Αυτές οι μέθοδοι, προϋποθέτουν την ύπαρξη μιας διαθέσιμης 'απόλυτα σωστής' περίληψης (στην οποία μπορούμε να υπολογίσουμε την ακρίβεια και την ανάκληση). Μπορούμε να λάβουμε μια τέτοια περίληψη με αρκετούς τρόπους. Πιο συνηθέστερα, λαμβάνεται με τη βοήθεια διαφόρων ανθρώπων που παράγουν περιλήψεις, και στη συνέχεια βρίσκοντας ένα 'μέσο όρων' αυτών. Αυτή η μέθοδος όμως είναι συνήθως προβληματική.

Μέθοδοι βασιζόμενοι σε περιεχόμενο

Αυτές οι μέθοδοι υπολογίζουν την ομοιότητα ανάμεσα σε δύο κείμενα σε ένα πιο λεπτομερές επίπεδο από αυτό των απλών προτάσεων. Η βασική μέθοδος συνίσταται από τον υπολογισμό της ομοιότητας μεταξύ του αρχικού κειμένου και της περίληψής του με χρήση της μετρικής ομοιότητας συνημιτόνου:

$$\cos(X, Y) = \frac{\sum x_i * y_i}{\sqrt{\sum (x_i)^2} * \sqrt{\sum (y_i)^2}},$$

όπου τα X και Y βασίζονται στο μοντέλο διανυσματικού χώρου.

Συσχέτιση ομοιότητας

Αφορά τον υπολογισμό της σχετικής μείωσης στο πληροφοριακό περιεχόμενο όταν γίνεται χρήση της περίληψης αντί του αρχικού κειμένου.

Αξιολόγηση βασισμένη σε εργασίες

Αυτές οι τεχνικές μετρούν την ανθρώπινη απόδοση χρησιμοποιώντας τις περιλήψεις για μια συγκεκριμένη εργασία (αφού έχουν παραχθεί οι περιλήψεις). Μπορούμε για παράδειγμα να μετρήσουμε την αποτελεσματικότητα της χρήσης περιλήψεων αντί των κειμένων για κατηγοριοποίηση αυτών. Αυτού του είδους η αξιολόγηση απαιτεί μια προ-κατηγοριοποιημένη συλλογή κειμένων (corpus).

2.5 Κατηγοριοποίηση Πληροφορίας

Η *κατηγοριοποίηση* αποτελεί μια από τις βασικές εργασίες *εξόρυξης δεδομένων*. Βασίζεται στην εξέταση των χαρακτηριστικών ενός νέου αντικειμένου (μη κατηγοριοποιημένο) το οποίο με βάση τα χαρακτηριστικά αυτά αντιστοιχίζεται σε ένα προκαθορισμένο σύνολο κλάσεων. Τα αντικείμενα που πρόκειται να κατηγοριοποιηθούν αναπαριστώνται γενικά από τις εγγραφές της βάσης δεδομένων και η διαδικασία της κατηγοριοποίησης αποτελείται από την ανάθεση κάθε εγγραφής σε κάποιες από τις προκαθορισμένες κατηγορίες. Ο στόχος της κατηγοριοποίησης κειμένου είναι η κατάταξη των κειμένων σε μια σταθερή σειρά προκαθορισμένων κατηγοριών. Κάθε κείμενο μπορεί να ανήκει σε καμία, ακριβώς μία, ή περισσότερες κατηγορίες.

Η εργασία της κατηγοριοποίησης χαρακτηρίζεται από έναν καλά καθορισμένο ορισμό των κατηγοριών και το σύνολο που χρησιμοποιείται για την εκπαίδευση του μοντέλου αποτελείται από προ-κατηγοριοποιημένα παραδείγματα. Η βασική εργασία είναι να δημιουργηθεί ένα μοντέλο το οποίο θα μπορούσε να εφαρμοστεί για να κατηγοριοποιεί δεδομένα που δεν έχουν ακόμα κατηγοριοποιηθεί. Στις περισσότερες περιπτώσεις, υπάρχει ένας περιορισμένος αριθμός (προκαθορισμένων) κατηγοριών και ο αλγόριθμος αναθέτει κάθε εγγραφή στην κατάλληλη κατηγορία. Για το σκοπό αυτό χρησιμοποιούνται κάποιες τεχνικές, τις οποίες μπορούμε να

κατατάξουμε σε δύο κατηγορίες. Η πρώτη χρησιμοποιεί *δέντρα αποφάσης* (Decision Trees) και η δεύτερη *Νευρωνικά δίκτυα* (*Neural Networks*). Και οι δύο στηρίζονται στην ιδέα της εκπαίδευσης (training) με τη βοήθεια ενός υποσυνόλου δεδομένων που ονομάζεται *σύνολο εκπαίδευσης* (*training set*). Το υποσύνολο αυτό επιλέγεται σαν αντιπροσωπευτικό δείγμα του συνολικού όγκου δεδομένων. Με την εφαρμογή της διαδικασίας εκπαίδευσης καθορίζονται κάποια πρότυπα για τις κατηγορίες δεδομένων. Έτσι, όταν προκύψει ένα νέο στοιχείο μπορεί εύκολα να κατηγοριοποιηθεί.

2.5.1 Αλγόριθμοι για κατηγοριοποίηση πληροφορίας

Η κατηγοριοποίηση χαρακτηρίζεται από ένα καλά καθορισμένο σύνολο κατηγοριών καθώς και ένα σύνολο από κατηγοριοποιημένα (pre-classified) παραδείγματα. Αντίθετα, η διαδικασία συσταδοποίησης δεν στηρίζεται σε προκαθορισμένες κατηγορίες ή παραδείγματα. Γενικά, ο στόχος της διαδικασίας κατηγοριοποίησης είναι η δημιουργία ενός μοντέλου που θα μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δεδομένων των οποίων η κατηγοριοποίηση είναι άγνωστη. Πιο συγκεκριμένα, η κατηγοριοποίηση μπορεί να περιγραφεί ως μια διαδικασία δύο βημάτων:

1. Εκμάθηση (Learning). Σε αυτό το βήμα χτίζεται ένα μοντέλο περιγράφοντας ένα προκαθορισμένο σύνολο από κατηγορίες δεδομένων. Τα δεδομένα εκπαίδευσης (training data) αναλύονται από έναν αλγόριθμο κατηγοριοποίησης για να κατασκευάσουν στη συνέχεια το μοντέλο. Τα στοιχεία που αποτελούν το σύνολο κατάρτισης επιλέγονται τυχαία από έναν πληθυσμό δεδομένων και ανήκουν σε μία από τις προκαθορισμένες κατηγορίες. Δεδομένου ότι η κατηγορία των δειγμάτων εκπαίδευσης είναι γνωστή, αυτό το βήμα είναι επίσης γνωστό σας '*εποπτεύομενη μάθηση*' (*supervised learning*).

2. Κατηγοριοποίηση (Classification). Σε αυτό το βήμα χρησιμοποιούνται τα δοκιμαστικά δεδομένα (data set) για να υπολογίσουν την ακρίβεια του μοντέλου. Υπάρχουν διάφορες μέθοδοι για να εκτιμηθεί η ακρίβεια του κατηγοριοποιητή. Τα δεδομένα εκπαίδευσης επιλέγονται τυχαία και είναι ανεξάρτητα. Το μοντέλο κατηγοριοποιεί κάθε ένα από τα δοκιμαστικά παραδείγματα (training samples). Στη συνέχεια η κατηγορία που ανήκουν τα δεδομένα με βάση το σύνολο δοκιμαστικών δεδομένων συγκρίνεται με την πρόβλεψη που έκανε το μοντέλο για την κατηγορία. Η ακρίβεια του μοντέλου σε ένα καθορισμένο σύνολο δεδομένων δοκιμής είναι το ποσοστό των δειγμάτων δοκιμής που κατηγοριοποιήθηκαν σωστά από το υπό εκπαίδευση μοντέλο. Εάν η ακρίβεια θεωρείται ως αποδεκτή, το μοντέλο μπορεί πλέον να χρησιμοποιηθεί για να κατηγοριοποιήσει και τα μελλοντικά δείγματα δεδομένων, των οποίων η κατηγοριοποίηση είναι άγνωστη.

Bayesian κατηγοριοποίηση

Η *Bayesian* κατηγοριοποίηση βασίζεται στη στατιστική θεωρία κατηγοριοποίησης του Bayes. Ο στόχος είναι να κατηγοριοποιηθεί ένα δείγμα X σε μια από τις δεδομένες κατηγορίες $C_1, C_2, ..., C_n$ χρησιμοποιώντας ένα μοντέλο πιθανότητας που ορίζεται σύμφωνα με τη θεωρία Bayes. Κάθε κατηγορία χαρακτηρίζεται από μια εκ των προτέρων πιθανότητα (a priori probability) παρατήρησης της κλάσης C_i. Επίσης, υποθέτουμε ότι το δεδομένο δείγμα Q ανήκει σε μια κλάση C_i, με την υπό συνθήκη συνάρτηση πυκνότητας πιθανότητας: $p(X/C_i) \in [0, 1]$. Κατόπιν, χρησιμοποιώντας τους ανωτέρω ορισμούς και βασιζόμενοι στη θεωρία Bayes, καθορίζουμε την εκ των υστέρων (posterior) πιθανότητα $p(c_i/x)$ ως:

$$p(c_i|X) = \frac{p(X|C_i)p(c_i)}{p(X)}$$

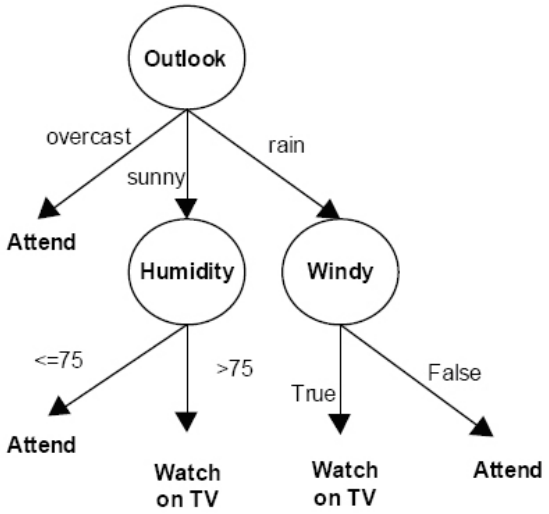
Ο απλούστερος Bayesian κατηγοριοποιητής είναι ο *Naive Bayesian*. Αυτός υποθέτει ότι η επίδραση ενός γνωρίσματος σε μια δεδομένη κατηγορία είναι ανεξάρτητη από τις τιμές των άλλων γνωρισμάτων. Αυτή η υπόθεση γίνεται για να απλοποιήσει τους υπολογισμούς που εμπλέκονται και καλείται *υπό συνθήκη ανεξαρτησία κατηγορίας*. Παρότι αυτή η υπόθεση δεν ισχύει συνήθως σε πραγματικά δεδομένα, ο αλγόριθμος είναι αρκετά αποτελεσματικός.

Ένας άλλος Bayesian κατηγοριοποιητής είναι τα Bayesian Belief Networks, τα οποία προσδιορίζουν τις συνδεδεμένες υπό συνθήκη κατανομές πιθανότητας στοχεύοντας στο να λάβουν υπόψη τις εξαρτήσεις που μπορούν να υπάρξουν μεταξύ των μεταβλητών.

Δέντρα απόφασης

Τα *δέντρα απόφασης* είναι μια από τις ευρέως χρησιμοποιούμενες τεχνικές για την κατηγοριοποίηση και την πρόβλεψη. Ένα δέντρο απόφασης κατασκευάζεται με βάση ένα σύνολο εκπαίδευσης προκατηγοριοποιημένων δεδομένων. Κάθε ένας από τους εσωτερικούς κόμβους του δέντρου απόφασης προσδιορίζει τον έλεγχο ενός γνωρίσματος και κάθε κλειδί που 'κατεβαίνει' από εκείνον τον κόμβο αντιστοιχεί σε μία από τις πιθανές τιμές για το συγκεκριμένο γνώρισμα. Επίσης, κάθε φύλλο αντιστοιχεί σε μια από τις κατηγορίες που έχουν ορισθεί.

Η διαδικασία για την κατηγοριοποίηση ενός νέου δείγματος με βάση ένα δέντρο απόφασης είναι η ακόλουθη: ξεκινώντας από τη ρίζα του δέντρου και εξετάζοντας τα γνωρίσματα που καθορίζονται από τον κόμβο αυτό προσδιορίζονται διαδοχικά οι εσωτερικοί κόμβοι που θα επισκεφθούμε έως ότου καταλήξουμε σε ένα φύλλο. Σε κάθε εσωτερικό κόμβο εξετάζεται εάν το δείγμα ικανοποιεί το συγκεκριμένο κόμβο. Η έκβαση αυτής της δοκιμής σ' έναν εσωτερικό κόμβο καθορίζει το κλαδί που θα διασχίσουμε στη συνέχεια καθώς και τον επόμενο κόμβο που θα επισκεφθούμε. Η κατηγορία του υπό μελέτη δείγματος είναι η κατηγορία του τελικού κόμβου ο οποίος αντιστοιχεί σε φύλλο του δέντρου.



Σχήμα 2.5: Δέντρο Απόφασης.

Διάφοροι αλγόριθμοι κατασκευής των δέντρων απόφασης έχουν αναπτυχθεί κατά τη διάρκεια των τελευταίων ετών. Μερικοί από τους πιο γνωστούς είναι οι:

- ID3
- C4.5
- SPRINT
- SLIQ
- CART
- RainForest

Γενικά, οι περισσότεροι από τους αλγορίθμους έχουν δύο διακριτές φάσεις: τη φάση οικοδόμησης και τη φάση περικοπής. Στη φάση οικοδόμησης, το σύνολο των δεδομένων εκπαίδευσης χωρίζεται κατ' επανάληψη μέχρις ότου όλα τα δείγματα σ' ένα τμήμα να ανήκουν στην ίδια κατηγορία. Το αποτέλεσμα είναι ένα δέντρο που κατηγοριοποιεί κάθε στοιχείο του συνόλου εκπαίδευσης. Ωστόσο, το δέντρο που έχει κατασκευάζεται μπορεί να είναι ευαίσθητο στις στατιστικές παρατυπίες του συνόλου κατάρτισης. Κατά συνέπεια, οι περισσότεροι από τους αλγορίθμους εκτελούν μια φάση περικοπής μετά από τη φάση κατασκευής του δέντρου,

στην οποία οι κόμβοι περικόπτονται για να αποτραπούν οι επικαλύψεις και για να δημιουργηθεί ένα δέντρο με υψηλότερη ακρίβεια. Οι διάφοροι αλγόριθμοι κατασκευής δέντρων απόφασης χρησιμοποιούν διαφορετικούς αλγορίθμους για την επιλογή του κριτηρίου ελέγχου για την κατηγοριοποίηση ενός συνόλου δεδομένων. Ένας από τους πιο πρόσφατους αλγορίθμους, ο *CLS*, εξετάζει όλα τα δυνατά δέντρα αποφάσεων σ' ένα συγκεκριμένο βάθος και στη συνέχεια επιλέγει τον έλεγχο που ελαχιστοποιεί το υπολογιστικό κόστος κατηγοριοποίησης ενός στοιχείου. Ο ορισμός αυτού του κόστους αποτελείται από το κόστος καθορισμού των τιμών των χαρακτηριστικών για έλεγχο καθώς και το κόστος λανθασμένης κατηγοριοποίησης.

Οι αλγόριθμοι *ID3* και *C4.5* βασίζονται σε μια στατιστική ιδιότητα, καλούμενη κέρδος πληροφορίας (information gain), προκειμένου να επιλέξουμε το γνώρισμα που θα ελέγξουμε σε κάθε κόμβο του δέντρου. Ο ορισμός του μέτρου βασίζεται στην εντροπία, η οποία χαρακτηρίζει την καθαρότητα μιας αφηρημένης επιλογής των δειγμάτων. Εναλλακτικά οι αλγόριθμοι όπως ο SLIQ, SPRINT επιλέγουν το γνώρισμα που θα ελεγχθεί με βάση το δείκτη *GINI* και όχι το μέτρο εντροπίας. Το καλύτερο γνώρισμα για τον έλεγχο δίνει και τη χαμηλότερη τιμή για τον δείκτη GINI.

Νευρωνικά δίκτυα

Μια άλλη προσέγγιση της κατηγοριοποίησης που χρησιμοποιείται σε πολλές εφαρμογές εξόρυξης γνώσης για πρόβλεψη και κατηγοριοποίηση βασίζεται στα *νευρωνικά δίκτυα*. Οι μέθοδοι αυτής της προσέγγισης χρησιμοποιούν τα νευρωνικά δίκτυα για να κατασκευάσουν ένα μοντέλο κατηγοριοποίησης ή πρόβλεψης. Τα κύρια βήματα της διαδικασίας είναι:

- Αναγνώριση των χαρακτηριστικών εισόδου και εξόδου.
- Κατασκευή ενός δικτύου με την κατάλληλη τοπολογία.
- Επιλογή του σωστού συνόλου εκπαίδευσης.
- Εκπαίδευση του δικτύου με βάση ένα αντιπροσωπευτικό σύνολο δεδομένων. Τα δεδομένα πρέπει να απεικονίζονται με τέτοιον τρόπο ώστε να μεγιστοποιηθεί η δυνατότητα του δικτύου να αναγνωρίζει πρότυπα.
- Έλεγχος του δικτύου χρησιμοποιώντας ένα σύνολο ελέγχου το οποίο είναι ανεξάρτητο από το σύνολο εκπαίδευσης.

Το μοντέλο που παράγεται από το δίκτυο εφαρμόζεται για να προβλέψει τις κατηγορίες των μη κατηγοριοποιημένων δειγμάτων.

Τα νευρωνικά δίκτυα αποτελούνται από 'νευρώνες' με βάση τη νευρωνική δομή του εγκεφάλου. Επεξεργάζονται τα στοιχεία ένα κάθε φορά και 'μαθαίνουν' συγκρίνοντας την κατηγοριοποίησή τους για μια εγγραφή (που, στην έναρξη, είναι κατά ένα μεγάλο μέρος αυθαίρετη) με τη γνωστή πραγματική κατηγοριοποίηση της εγγραφής. Τα λάθη από την αρχική κατηγοριοποίηση της πρώτης εγγραφής επανατροφοδοτούνται στο δίκτυο, και χρησιμοποιούνται για να τροποποιήσουν τον αλγόριθμο δικτύων τη δεύτερη φορά. Η διαδικασία αυτή συνεχίζεται επαναληπτικά.

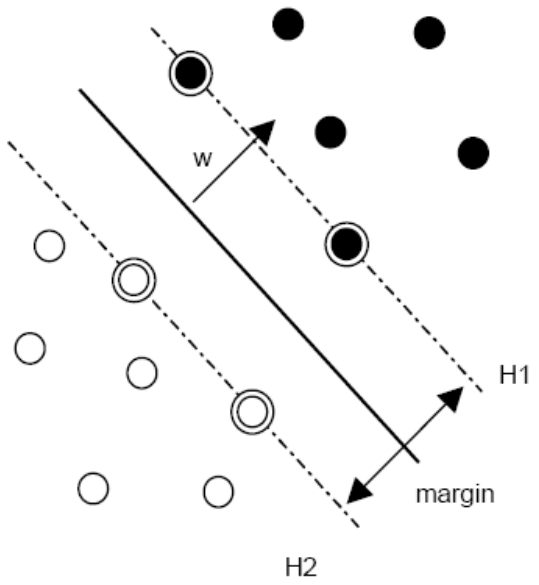
Κοντινότεροι γείτονες (*NearestNeighbors - NN*)

Η τεχνική των *κοντινότερων γειτόνων* είναι μια απλή προσέγγιση του προβλήματος της κατηγοριοποίησης. Σύμφωνα με αυτή, ένα νέο στοιχείο κατηγοριοποιείται χρησιμοποιώντας την πλειοψηφία μεταξύ των κατηγοριών από τα K παραδείγματα που είναι τα πιο κοντινά σ' αυτό που δίνεται να κατηγοριοποιηθεί. Μια τέτοια μέθοδος παράγει συνεχείς και επικαλυπτόμενες, παρά σταθερές γειτονιές. Επιπρόσθετα, έχει αποδειχθεί ότι ένας NN κανόνας έχει ασυμπτωτικό ποσοστό σφάλματος που είναι δύο φορές το ποσοστό σφάλματος Bayes, ανεξάρτητα από το μέτρο απόστασης που χρησιμοποιείται.

Η τεχνική NN έχει μειονεκτήματα σε χώρους υψηλών διαστάσεων. Η αυστηρή πόλωση μπορεί να εισαχθεί στην τεχνική NN όταν υπάρχει ένας πεπερασμένος αριθμός από τα παραδείγματα σε χώρο υψηλών διαστάσεων.

Support Vector Machines

Τα *SVMs* είναι μια καινούρια μέθοδος κατηγοριοποίησης η οποία προτάθηκε από τον Vapnik, και έχει ήδη αποκτήσει μεγάλη δημοσιότητα. Στην πιο απλή του μορφή, ένα SVM ορίζεται σαν έναν υπερεπίπεδο που δύναται να διαχωρίσει ένα σύνολο θετικών από ένα σύνολο αρνητικών στοιχεία που αφορούν μια συγκεκριμένη κατηγορία. Αυτό φαίνεται και στο παρακάτω σχήμα όπου υποθέτοντας ότι οι μαύρες κουκίδες αφορούν τα θετικά στοιχεία και οι άσπρες τα αρνητικά στοιχεία, ορίζεται με τη βοήθεια του SVM ένα μέγιστο υπερεπίπεδο που αποτελεί το διαχωριστικό ανάμεσα στα στοιχεία.



Σχήμα 2.6: Γραμμικά χωρισμένα υπερεπίπεδα.

Στη γραμμική μορφή του αλγορίθμου, το περιθώριο μεταξύ των στοιχείων μπορεί να οριστεί σαν η απόσταση του υπερεπιπέδου από τα κοντινότερα θετικά και αρνητικά στοιχεία. Η μεγιστοποίηση αυτού του περιθωρίου μπορεί να αποτελέσει ένα πρόβλημα βελτιστοποίησης. Φυσικά τα περισσότερα παραδείγματα δε μπορούν να διαχωριστούν με τη χρήση της γραμμικής μορφής του αλγορίθμου γι' αυτό χρησιμοποιούνται πίνακες προκειμένου να υπολογιστούν τα περιθώρια και οι αποστάσεις. Οι αλγόριθμοι για SVM έχουν αποδειχθεί ότι έχουν καλή γενικά απόδοση ακόμα και σε δύσκολα προβλήματα κατηγοριοποίησης μερικά από τα οποία είναι η αναγνώριση γραφικού χαρακτήρα, η αναγνώριση προσώπου, η κατηγοριοποίηση κειμένων. Η απλή γραμμική μορφή έχει πολύ καλή απόδοση, υφίσταται γρήγορη εκμάθηση και παράλληλα μπορεί να κατηγοριοποιεί εξαιρετικά γρήγορα. Περισσότερα στοιχεία για το SVM μπορούν να βρεθούν στο [42].

Ασαφής κατηγοριοποίηση (*Fuzzy Classification*)

Οι προηγούμενες μέθοδοι κατηγοριοποίησης παράγουν μια αυστηρή κατηγοριοποίηση με την έννοια ότι ένα αντικείμενο είτε ανήκει σε μια κατηγορία είτε όχι. Αυτό σημαίνει ότι όλα τα αντικείμενα θεωρούνται ότι ανήκουν σε μια κατηγορία με τον ίδιο βαθμό πίστης. Επιπλέον, οι κατηγορίες θεωρούνται ως μη επικαλυπτόμενες. Είναι προφανές ότι δεν λαμβάνεται υπόψη η έννοια της αβεβαιότητας σε αυτές τις μεθόδους.

Μια εναλλακτική προσέγγιση στο πρόβλημα αφορά την ασαφή κατηγοριοποίηση η οποία βασίζεται στην ασαφή λογική. Η βασική ιδέα είναι η εξαγωγή των ασαφών κανόνων προκειμένου να αναγνωριστεί κάθε κατηγορία δεδομένων. Οι μέθοδοι εξαγωγής κανόνων βασίζονται στον υπολογισμό των ομάδων (κατηγοριών) στα δεδομένα και κάθε κατηγορία που ορίζεται αντιστοιχεί σε έναν ασαφή κανόνα που συσχετίζει μια περιοχή στο χώρο εισόδου με μια κατηγορία εξόδου. Κατά συνέπεια, για κάθε κατηγορία C_i καθορίζεται το κέντρο των ομάδων έτσι ώστε να προκύψει ένας κανόνας της μορφής:
if {input is near x_i} then class is C_i

Κατόπιν για ένα δεδομένο διάνυσμα εισόδου x, το σύστημα ορίζει το βαθμό ικανοποίησης κάθε κανόνα και τα συμπεράσματα του κανόνα με τον υψηλότερο βαθμό ικανοποίησης επιλέγεται ως η έξοδος του ασαφούς συστήματος. Έτσι η προσέγγιση χρησιμοποιεί ασαφή λογική για να καθορίσει την καλύτερη κατηγορία μέσα στην οποία ένα δεδομένο μπορεί να κατηγοριοποιηθεί, αλλά το τελικό αποτέλεσμα είναι η κατηγοριοποίηση κάθε στοιχείου σε μια από τις κατηγορίες.

Παραγωγή κανόνων κατηγοριοποίησης

Η γνώση που παράγεται κατά τη διάρκεια της διαδικασίας της κατηγοριοποίησης μπορεί να εξαχθεί και να αναπαρασταθεί υπό τη μορφή κανόνων. Μια κοινή προσέγγιση της κατηγοριοποίησης είναι τα δέντρα απόφασης. Σε αυτή την περίπτωση τα πρότυπα γνώσης που εξάγονται περιγράφονται υπό τη μορφή ενός δέντρου. Ωστόσο οι κανόνες είναι ευκολότερα αντιληπτοί από τους ανθρώπους, ιδιαίτερα εάν το δέντρο είναι πολύ μεγάλο.

2.6 Προσωποποίηση στο χρήστη

Η προσωποποίηση στο χρήστη είναι διαδικασία κατά την οποία τα αποτελέσματα που εμφανίζονται τελικά στο χρήστη προσαρμόζονται προκειμένου να ανταποκρίνονται στις ανάγκες του. Πιο συγκεκριμένα, τα στάδια της προσωποποίησης αφορούν τον εντοπισμό άρθρων τα οποία ενδιαφέρουν το χρήστη και παρουσίασή τους με τέτοιον τρόπο ώστε να ταιριάζουν στις ανάγκες του χρήστη. Παράλληλα, η προσωποποίηση περνάει σε ένα ακόμη επίπεδο αφού λαμβάνει υπ' όψιν και τις δυνατότητες απεικόνισης της τελικής συσκευής του χρήστη ώστε να στέλνεται κάθε φορά το κατάλληλο μέγεθος περιλήψεων για την όσο το δυνατόν πιο εύληπτη απεικόνιση στη συσκευή. Το πρόβλημα που τίθεται είναι η εύρεση ενός 'έξυπνου' αλγορίθμου ο οποίος θα μπορεί να αξιοποιεί όλες τις πληροφορίες που μπορούν προέρχονται από τον χρήστη προκειμένου να του επιστραφούν όσο το δυνατόν καλύτερα και ποιοτικότερα αποτελέσματα.

2.7 Συμμετοχή του χρήστη στις διαδικασίες του συστήματος

Ο χρήστης είναι αυτός που δέχεται την τελική πληροφορία και αυτός που ουσιαστικά διαμορφώνει την πληροφορία για τον εαυτό του. Αυτό σημαίνει πως ο χρήστης θα πρέπει να είναι αναπόσπαστο κομμάτι του συστήματος. Θα πρέπει να είναι σε θέση να διαμορφώσει διαδικασίες του πυρήνα του συστήματος όπως είναι η κατηγοριοποίηση και η εξαγωγή περίληψης. Στα περισσότερα συστήματα τα οποία αντιμετωπίστηκαν κατά τη διάρκεια της μελέτης για τη συγκεκριμένη εργασία, παρατηρήθηκε πως ο χρήστης συμμετέχει μόνο στα επιτελικά στάδια των συστημάτων ενώ έχουν ήδη εκτελεστεί τα βασικά βήματα του πυρήνα των μηχανισμών. Η συμμετοχή του χρήστη στις διαδικασίες πυρήνα ενός large scale συστήματος είναι επίπονη διαδικασία η οποία απαιτεί αλγορίθμους που θα μπορούν να εκτελούνται αποδοτικά σε πραγματικό χρόνο προκειμένου ο χρήστης να διαμορφώνει όχι μόνον τα τελικά αποτελέσματα που εμφανίζονται σε αυτόν αλλά και συγκεκριμένες διαδικασίες ολόκληρου του συστήματος.

2.8 Αξιοποίηση Πληροφορίας

Η πληροφορία που ανακτάται τόσο από το μηχανισμό εξόρυξης όσο και από το μηχανισμό κατηγοριοποίησης είναι υπέρογκη. Αρκεί να φανταστεί κάποιος ότι από 100 τυχαίες ηλεκτρονικές διευθύνσεις εξάγονται 90-95 κείμενα, από τα οποία λαμβάνουμε 2000 διακριτές λέξεις (πριν τη διαδικασία του stemming) και από τις οποίες προκύπτουν 8000-10000 συσχετίσεις κείμενο-λέξη-βάρος. Για το λόγο αυτό θα πρέπει να υπάρχει ένας ισχυρός μηχανισμός που να είναι σε θέση να αξιοποιήσει τη συγκεκριμένη πληροφορία και να μπορεί να βελτιώσει τους τρόπους που γίνονται ερωτήματα στη βάση και προσθήκες νέων εγγραφών.

Αυτό που θα πρέπει να μας απασχολήσει περισσότερο για το συγκεκριμένο σύστημα είναι να δημιουργηθεί ένας μηχανισμός διαχείρισης της πληροφορίας. Η πληροφορία δε θα πρέπει να είναι στάσιμη. Συνεχώς θα ανανεώνεται, και θα πρέπει ανελλιπώς να διαγράφονται ή να τροποποιούνται τα στοιχεία τα οποία δε

συγκεντρώνουν το ενδιαφέρον των χρηστών του συστήματος. Τα κείμενα αλλάζουν διαρκώς, το ίδιο και οι προτιμήσεις των χρηστών.

Μέσα από την πλήρη καταγραφή των προτιμήσεων του χρήστη, ο μηχανισμός θα είναι σε θέσει να συνδυάσει αυτές τις πληροφορίες μαζί με αυτές από την κατηγοριοποίηση, για να παράγει προσωποποιημένη μορφή περίληψης άρθρων του διαδικτύου τα οποία 'ταιριάζουν' στον εκάστοτε χρήστη, τόσο βάσει των προτιμήσεών του, όσο και βάσει των δυνατοτήτων της συσκευής που έχει στην κατοχή του.

2.9 Προφίλ χρήστη για δυναμικά περιβάλλοντα

Ένα πολύ σημαντικό στοιχείο της εργασία είναι το προφίλ χρήστη σε δυναμικό περιβάλλον. Είναι το στοιχείο που χαρακτηρίζει την πύλη ποιοτικού περιεχομένου και είναι ένα από τα βασικά στοιχεία που δίνουν νόημα στη λέξη ποιότητα της πύλης.

Το δυναμικό περιβάλλον της πύλης θα δίνει τη δυνατότητα πρόσβασης σε πληροφορία η οποία ενδιαφέρει το χρήστη, καταργώντας τα περιθώρια εμφάνισης ανεπιθύμητων αποτελεσμάτων. Προκειμένου να γίνει κατανοητό θα πρέπει να προσδιοριστεί ο όρος προφίλ χρήστη.

Στο άκουσμα του όρου προφίλ χρήστη θα περίμενε κανείς να έρθει αντιμέτωπος με προσωπικά στοιχεία του χρήστη (όνομα, επώνυμο κλπ.). Όσο κι αν ακούγεται παράξενο, σε ένα δυναμικό περιβάλλον ίσως δεν έχει και τόσο μεγάλη σημασία ο προσδιορισμός του χρήστη σαν φυσικό πρόσωπο αλλά περισσότερο σαν χρήστης του διαδικτύου. Βασικός στόχος της δημιουργίας του προφίλ ενός χρήστη είναι να προσδιοριστεί με όσο μεγαλύτερη ακρίβεια η δράση του φυσικού προσώπου όταν έρχεται αντιμέτωπος με το διαδίκτυο. Είναι μεγάλο επίτευγμα να μπορεί κανείς να προσδιορίσει την επόμενη κίνηση που θα πραγματοποιήσει ο χρήστης (π. χ. ποιο σύνδεσμο θα ακολουθήσει στην επόμενη κίνηση). Ακούγεται σαν παιχνίδι πρόβλεψης και ίσως θα μπορούσε να παρομοιαστεί με κάτι τέτοιο. Ωστόσο είναι κάτι πιο σύνθετο και βασίζεται σε μία πληθώρα στοιχείων. Τι ερωτήματα πραγματοποιεί ο χρήστης, ποιες σελίδες επισκέπτεται πιο συχνά από τα αποτελέσματα που του εμφανίζονται, τι έχει δηλώσει σαν 'αγαπημένες κατηγορίες' αποτελούν μερικά από τα βασικά στοιχεία πάνω στα οποία βασίζεται η δημιουργία του προφίλ ενός χρήστη.

Στο συγκεκριμένο σύστημα, το ενδιαφέρον μας επικεντρώνεται στην αξιολόγηση που κάνει ο χρήστης όταν του παρουσιάζονται τα αποτελέσματα της αναζήτησής του. Ένα παράδειγμα θα ήταν αρκετό για να κατανοήσει κανείς το νόημα που έχει το 'δυναμικό προφίλ' στη συγκεκριμένη δικτυακή πύλη. Έστω ένας χρήστης του διαδικτύου που χρησιμοποιεί τη συγκεκριμένη δικτυακή πύλη και επιθυμεί να βλέπει καθημερινά τα περιεχόμενα της κατηγορίας business. Το προφίλ έχει ήδη δημιουργηθεί και περιλαμβάνει την πολύ γενική κατηγορία βυσινεσς. Όταν παρουσιάζονται στο χρήστη αποτελέσματα (τίτλος άρθρου, μικρό απόσπασμα άρθρου), τότε ο χρήστης επιλέγει κάποιο ή κάποια αποτελέσματα για να τα εξετάσει περαιτέρω. Το κάθε κείμενο όμως αποτελείται, συν τοις άλλοις, και από κάποιες λέξεις-κλειδιά. Μόλις κάποιος χρήστης επιλέξει κάποιο κείμενο, οι λέξεις-κλειδιά που υπάρχουν στο συγκεκριμένο, αυτομάτως αποκτούν αξία για το συγκεκριμένο χρήστη και εισάγονται αυτόματα στο προφίλ του. Αυτή η πληροφορία είναι πολύ σημαντική προκειμένου το σύστημα να είναι σε θέση να κάνει μεγαλύτερη αξιολόγηση των κειμένων που θα παρουσιάσει στο χρήστη. Έτσι, την επόμενη φορά που ο χρήστης θα δει τα αποτελέσματα για την κατηγορία που επιθυμεί τα κείμενα θα είναι ταξινομημένα (και) βάσει των λέξεων-κλειδιών που έχουν τη μεγαλύτερη βαθμολογία για κάθε χρήστη. Με αυτό τον τρόπο αποκτά μεγαλύτερη αξία το κείμενο που περιέχει πολλές λέξεις-κλειδιά για ένα συγκεκριμένο χρήστη. Η συγκέντρωση των αποτελεσμάτων συνολικά για τους χρήστες μίας κατηγορίας μπορεί να οδηγήσει σε μεγαλύτερη διαβάθμιση κάθε κατηγορίας και δημιουργία εικονικών υποκατηγοριών που θα είναι χωρισμένες βάση της απόκρισης των χρηστών. Θεωρητικά ένα τέτοιο μοντέλο, εικονικής ουσιαστικά, κατηγοριοποίησης είναι πιο αποτελεσματικό από κάθε αλγοριθμικό μοντέλο καθώς η κατηγοριοποίηση δε γίνεται από τη μηχανή αλλά από τον άνθρωπο.

2.10 Συσκευές μικρού μεγέθους

Η ασύρματη πρόσβαση στον παγκόσμιο ιστό από τους φορητούς προσωπικούς ψηφιακούς βοηθούς (PDAs) είναι μια συναρπαστική και συνάμα ελπιδοφόρος προσθήκη στη χρήση του Ιστού. Συχνά, ξέρουμε ότι οι πληροφορίες που χρειαζόμαστε είναι online, αλλά δεν μπορούμε να έχουμε πρόσβαση σε αυτές, επειδή π. χ. δεν είμαστε κοντά στο γραφείο μας. Οι συσκευές μικρού μεγέθους που υποστηρίζουν υπηρεσίες

και περιβάλλοντα του Web, είναι σε γενικές γραμμές, ένα τέλειο μέσο για τέτοιες ανάγκες πληροφοριών ακριβώς όταν προκύπτουν.

Δυστυχώς, η πρόσβαση με χρήση συσκευών μικρού μεγέθους στον Παγκόσμιο Ιστό συνεχίζει να θέτει ορισμένες δυσκολίες στους χρήστες [79]. Η μικρή οθόνη δίνει γρήγορα αποτελέσματα και πληροφορία που συγχέουν τον χρήστη. Η εισαγωγή των πληροφοριών από τη άλλη, είναι εξαιρετικά αργή σε σχέση με έναν παραδοσιακό υπολογιστή, κυρίως λόγω του μικρού πληκτρολογίου που διαθέτουν αυτές οι συσκευές. Αυτή η κατάσταση οδηγεί τους χρήστες σε λάθη και σε χαμένο χρόνο. Η ταχύτητα κατεβάσματος επίσης είναι σημαντικά μικρότερη σε σχέση με τα παραδοσιακά ενσύρματα (ή και ασύρματα) δίκτυα υπολογιστών, ενώ παράλληλα το κόστος χρήσης της υπηρεσίας είναι τάξεις μεγέθους μεγαλύτερο. Τα προβλήματα αυτά μας οδηγούν στην επιλογή μιας λύσης η οποία θα παρέχει αξιόπιστη, ταχύτατη, μορφοποιημένη σύμφωνα με ευρέως αποδεκτά πρότυπα και μόνο χρήσιμη πληροφορία στον τελικό χρήστη.

2.10.1 *RSS*

Το *RSS - Real Simple Syndication*[31] είναι μια οικογένεια από σχήματα τροφοδότησης περιεχομένου στους χρήστες που χρησιμοποιούνται για να δημοσιεύουν συχνά ενημερωμένο περιεχόμενο όπως οι καταχωρήσεις blog, οι τίτλοι ειδήσεων ή τα podcasts. Ένα έγγραφο RSS, που καλείται επίσης και 'feed', περιέχει είτε μια περίληψη του περιεχομένου του σχετικού ιστοχώρου, είτε το πλήρες κείμενο. Το RSS καθιστά δυνατό για τους χρήστες να παρακολουθούν τους αγαπημένους ιστοχώρους τους με έναν αυτοματοποιημένο τρόπο χωρίς να απαιτείται η πλοήγηση σε αυτούς. Το περιεχόμενο RSS μπορεί να διαβαστεί χρησιμοποιώντας λογισμικό γνωστό και ως '*feed reader*' ή '*aggregator*'. Από τη στιγμή που ο χρήστης γίνεται συνδρομητής σε ένα feed, ο aggregator αναλαμβάνει να δέχεται τα νέα που προέρχονται από το feed ανά τακτά χρονικά διαστήματα. Υπάρχουν διάφορα πρότυπα RSS, RSS 2.0, RSS 1.0, RSS 0.91, όλα όμως χρησιμοποιούν μια XML δομή για τη σύνταξη των δεδομένων.

Η χρήση των προτύπων RSS βοηθάει τους χρήστες στην προσπάθεια τους να αξιολογήσουν το περιεχόμενο που διαβάζουν, μια εργασία που όμως δεν είναι αυτοματοποιημένη, αφού οι ίδιοι οι χρήστες είναι αυτοί που θα πρέπει να ξεχωρίζουν τελικά ποια από τα νέα που έρχονται από τα RSS feeds τους ενδιαφέρουν. Τα πράγματα γίνονται ακόμη δυσκολότερα όταν πολλά μεγάλα News Portals προσφέρουν μέσω των RSS καναλιών τους έναν τίτλο και ένα μικρό μόνο περιεχόμενο (π. χ. 1-2 προτάσεις) για κάθε νέο. Αποτέλεσμα των παραπάνω είναι ο χρήστης να πρέπει πράγματι να επισκεφθεί διάφορα *news portals* αναζητώντας το περιεχόμενο που τον ενδιαφέρει, μια κατάσταση εξαιρετικά ανεπιθύμητη για τους χρήστες συσκευών μικρού μεγέθους.

Ως απάντηση στα παραπάνω προβλήματα, ο μηχανισμός που αναπτύχθηκε επιχειρεί την διαρκή, έγκαιρη και έγκυρη τροφοδότηση των χρηστών συσκευών μικρού μεγέθους (και όχι μόνο) με χρήση RSS feed που περιέχει περιλήψεις νέων τα οποία προέρχονται από μεγάλα και γνωστά ειδησεογραφικά sites ανά τον κόσμο. Οι περιλήψεις αυτές περιέχουν κείμενο κατηγοριοποιημένο και προσωποποιημένο ανάλογα με τις προτιμήσεις του χρήστη και μπορούν να του παρέχουν μια ολοκληρωμένη ενημέρωση για τα θέματα που τον ενδιαφέρουν απλά και μόνο διαβάζοντας ένα RSS feed. Η ευκολία αυτή, που εκτιμάται θετικά, ιδιαίτερα σε συνθήκες κίνησης όπου μόνο συσκευές μικρού μεγέθους είναι διαθέσιμες, επεκτείνεται και σε κανονικούς υπολογιστές όπου διαφέρουν σημαντικά σε χαρακτηριστικά οθόνης. Κάθε χρήστης έχει οθόνη προβολής με διαφορετικές δυνατότητες, επομένως διαφορετική περίληψη πρέπει να λάβει ο χρήστης που βρίσκεται μπροστά σε ένα laptop με υψηλή ανάλυση οθόνης και διαφορετική εκείνος που χρησιμοποιεί ένα κινητό τηλέφωνο. Όλα αυτά τα ζητήματα αναλύονται διεξοδικά στις ενότητες που ακολουθούν.

ΚΕΦΑΛΑΙΟ 3

Σχετικές εργασίες

> Computers are useless. They can only give you answers.
>
> *Pablo Picasso, Spanish Cubist painter*

Στο παρών κεφάλαιο παρουσιάζεται η ερευνητική δραστηριότητα στο χώρο για τα θέματα με τα οποία καταπιάνεται η εργασία. Ακολουθεί ουσιαστικά μια συνοπτική παρουσίαση των σχετικών εργασιών για θέματα συλλογής, φιλτραρίσματος, προεπεξεργασίας δεδομένων, κατηγοριοποίησης και αυτόματης εξαγωγής περίληψης κειμένου. Τέλος γίνεται μια παρουσίαση των τρέχοντων ερευνητικών θεμάτων που έχουν να κάνουν με προσωποποίηση περιεχομένου στο χρήστη.

3.1 Συλλογή δεδομένων

Για τη συλλογή δεδομένων από το διαδίκτυο χρησιμοποιούνται οι ευρέως γνωστοί *crawlers*. Το πλήθος τους είναι αμέτρητο ενώ, αν εξαιρέσουμε τους εξειδικευμένους crawlers (focused crawlers) παρατηρούμε πως οι περισσότεροι έχουν σαν σκοπό να συλλέξουν όλες τις HTML σελίδες από τις οποίες απαρτίζεται ένας δικτυακός τόπος μαζί με τα βοηθητικά αρχεία (pdf, εικόνες, video, css, javascript) και ουσιαστικά να δημιουργήσουν ένα offline-instance του δικτυακού τόπου τον οποίο προσπελαύνουν. Οι crawlers που έχουν κατασκευαστεί για το διαδίκτυο αγγίζουν σε αριθμό τις μερικές χιλιάδες καθώς η κατασκευή τους είναι σχεδόν τετριμμένη. Στη συνέχεια θα παρουσιάσουμε συγκεκριμένου crawlers που αξίζουν προσοχής για τα ιδιαίτερα χαρακτηριστικά που παρουσιάζουν.

3.1.1 *WebCrawler*

Πρόκειται για έναν από τους πρώτους crawlers που κατασκευάστηκαν από τον Pinkerton το 1994 [110]. Βασίστηκε στη βιβλιοθήκη WWW προκειμένου να είναι σε θέση να κατεβάζει σελίδες από το διαδίκτυο ενώ χρησιμοποιούσε ένα δεύτερο πρόγραμμα προκειμένου να διαβάζει τα URL τα οποία πρέπει να προσπελάσει. Ο αλγόριθμος προσπέλασης ήταν κατά πλάτος αναζήτηση του γραφήματος μίας ιστοσελίδας σε συνδυασμό με αποφυγή των σελίδων που έχει ήδη επισκεφθεί. Ένα αξιοσημείωτο στοιχεία ήταν η δυνατότητα να ακολουθεί συγκεκριμένα μόνο links σε ένα δικτυακό τόπο - και όχι όλα - βάση του ερωτήματος που έθετε ο χρήστης. Ήταν κάτι σαν ένας crawler πραγματικού χρόνου που φυσικά μπορούσε να ανταποκριθεί πλήρως λόγω του μικρού μεγέθους που είχε το διαδίκτυο.

3.1.2 *Google Crawler*

Ένας από τους πιο σημαντικούς crawlers που κατασκευάστηκαν και διατηρούνται ακόμα και σήμερα, με σημαντικές βέβαια βελτιώσεις είναι ο *Google Crawler* των Brin και Page, 1998 [47]. Βασίζεται στις

γλώσσες προγραμματισμού C++ και Python και παρουσιάζει εξαιρετικά μεγάλη πολυπλοκότητα. Επειδή η χρήση των σελίδων που κατέβαζε ο crawler προοριζόταν για εκτενή αναζήτηση μέσα σε σειρές από κείμενα, ο συγκεκριμένος crawler βασίστηκε στη διαδικασία indexing. Στο μηχανισμό υπάρχει ένας URL εξυπηρετητής που αποστέλλει λίστες με URL προς τους crawlers του συστήματος οι οποίοι λειτουργούν παράλληλα. Οι crawlers εξάγουν από τις σελίδες το κείμενο αλλά και όσα URLs εντοπίζουν. Αυτά στέλνονται πίσω στον URL εξυπηρετητή για έλεγχο και σε περίπτωση που δεν τα έχει επισκεφθεί ποτέ ο crawler προστίθενται στη λίστα του εξυπηρετητή.

3.1.3 *Mercator*

Ο *Mercator* [72] είναι ένας κατανεμημένος τμηματοποιημένος web crawler γραμμένος εξ' ολοκλήρου σε γλώσσα προγραμματισμού Java. Η τμηματοποίηση του προκύπτει από τη χρήση δύο διαφορετικών πρωτοκόλλων.

- Protocol modules
  - Τα τμήματα πρωτοκόλλων είναι υπεύθυνα για την ομαλή σύνδεση του μηχανισμού στις σελίδες και για την εξασφάλιση πως ο μηχανισμός θα είναι σε θέση να 'κατεβάσει' τη σελίδα.
- Processing modules
  - Από την άλλη μεριά τα τμήματα επεξεργασίας είναι αυτά που αφορούν την ανάλυση της σελίδας και την εξαγωγή του κειμένου και συνδέσμων από αυτή. Η απλή διαδικασία επεξεργασίας περιλαμβάνει ανάλυση της σελίδας και εξαγωγή των συνδέσμων που αυτή περιέχει ενώ σε μία πιο σύνθετη μορφή της περιλαμβάνει αλγορίθμους για την αποτελεσματική εξαγωγή του κειμένου.

3.1.4 *WebFountain*

Πρόκειται για έναν κατανεμημένο τμηματικό crawler παραπλήσιο του mercator, με τη διαφορά ότι είναι γραμμένος σε C++. Περιλαμβάνει έναν κεντρικό μηχανισμό και μία σειρά από "ant" (μερμήγκι) μηχανισμούς [59]. Πρόκειται δηλαδή για το ρυθμιστή της κατάστασης και τους εργάτες. Ο μηχανισμός αυτός περιέχει στοιχεία που τον κάνουν πολύ φιλικό προς τις σελίδες που επισκέπτεται. Σκοπός του είναι η διατήρηση ενός off-line instance του διαδικτύου. Αυτό έχει σαν αποτέλεσμα, μία από τις μετρικές τις οποίες προσμετρά ο συγκεκριμένος μηχανισμός να είναι το κατά πόσο η σελίδες που διαθέτει ανταποκρίνονται στις πραγματικές σελίδες που βρίσκονται on-line στους δικτυακούς τόπος και όχι απλά μία παλαιότερη έκφανσή τους. Για να πετύχει μεγαλύτερο freshness όπως ονομάζεται η συγκεκριμένη μετρική χρησιμοποιεί διαφορετική συχνότητα επίσκεψης στις σελίδες που έχει αποθηκευμένες στη βάση δεδομένων του.

3.1.5 *WebRACE*

Πρόκειται για έναν crawler ο οποίος είναι γραμμένος σε Java και αποτελεί ένα κομμάτι ενός γενικότερου συστήματος που ονομάζεται eRACE [135]. Το συγκεκριμένο σύστημα λαμβάνει εντολές από τους τελικούς χρήστες για να ξεκινήσει να κατεβάσει σελίδες και συμπεριφέρεται σαν proxy server. Το σύστημα μπορεί να εξυπηρετήσει και αιτήσεις για αλλαγές στοιχείων σε σελίδες: μόλις μία σελίδα αλλάξει, τότε ο crawler την ξανακατεβάζει και ειδοποιεί τον τελικό χρήστη που ενδιαφέρεται πως η σελίδα έχει αλλάξει και πως πλέον στον proxy είναι αποθηκευμένη μία νέα σελίδα. Το πιο σημαντικό στοιχείο του συγκεκριμένου crawler είναι η χαρακτηριστική διαφορά που παρουσιάζει συγκριτικά με όσους crawlers έχουμε δει. Στο συγκεκριμένο crawler δεν υπάρχει ένα feed URL από το οποίο θα ξεκινήσει να αναζητά σελίδες. Το URL feed είναι δυναμικό και διαμορφώνεται από τα ερωτήματα των χρηστών. Μετά τη χρήση του καταστρέφεται και ο μηχανισμός βρίσκεται σε αναμονή μέχρι να του δοθεί κάποιο νεότερο ερώτημα.

3.1.6 *Ubicrawler*

Ο *Ubicrawler* [45] είναι ένας κατανεμημένος crawler γραμμένος σε Java και δε διαθέτει κεντρικοποιημένη διαδικασία. Είναι κατασκευασμένος από έναν αριθμό από όμοιους "agents" και μία συνάρτηση ανάθεση που αναθέτει σε κάθε agent κάποια εργασία. Οι agents δεν επικοινωνούν μεταξύ τους άμεσα αλλά όλες οι

διαδικασίες διευθετούνται από την κεντρική συνάρτηση ανάθεσης. Καμία σελίδα δεν προσπελαύνεται διπλή φορά καθώς κάθε agent φροντίζει να ενημερώσει για τις σελίδες που έχει επισκευθεί εκτός και αν κάποιος από τους agents καταστραφεί. Πρόκειται για έναν πολύ σταθερό crawler, σχεδιασμένο με τέτοιον τρόπο ώστε να πετυχαίνει μέγιστη κλιμάκωση και μικρή ευαισθησία σε σφάλματα.

3.1.7 *Crawlers* Ανοιχτού Κώδικα

Μία σειρά από crawlers ανοιχτού κώδικα διανέμονται ελεύθερα στο διαδίκτυο. Κυρίως είναι προϊόντα κάποιου ιδιώτη που κατασκευάζονται για να καλύψουν συγκεκριμένες ανάγκες που έχουν οι τελικοί χρήστες, ανάγκες που συχνά δεν καλύπτονται από τους εμπορικούς crawlers. Η χρήση τους έχει συνήθως ως εξής. Κάποιος χρήστης που δεν καλύπτεται από έναν εμπορικό crawler λαμβάνει τον κώδικα ενός open source συστήματος και το αλλάζει με σκοπό να το φέρει στα μέτρα του. Συνήθως οι open source crawlers δεν έχουν εξειδικευμένες λειτουργικότητες ωστόσο προσφέρονται στους τελικούς χρήστες οι οποίοι μπορούν να τους τροποποιήσουν ελεύθερα.

Μερικά παραδείγματα από crawlers ανοιχτού κώδικα ακολουθούν

- GNU Wget [12]
- Heritrix [14]
- ht://Dig [15]
- HTTrack [16]
- Larbin [20]
- Methabot [21]
- Nutch [25]
- WebSPHINX [34]
- WIRE - Web Information Retrieval Environment [33]

3.2 Φιλτράρισμα δεδομένων

Η διαδικασία της εξαγωγής κειμένου για τον σκοπό για τον οποίο χρησιμοποιείται στη συγκεκριμένη εργασία ξεφεύγει από το σκοπό που έχουν οι ελάχιστες εμπορικές εφαρμογές. Έτσι η εξαγωγή χρήσιμου κειμένου από HTML σελίδες αποτελεί αντικείμενο έρευνας ενώ η εξαγωγή όλου του κειμένου μίας HTML σελίδας αποτελεί μία τετριμμένη διαδικασία.

Η εξαγωγή κειμένου από HTML σελίδες είναι μία απλοϊκή διαδικασία η οποία βασίζεται στην αφαίρεση των HMTL ταγς και στη διατήρηση του υπόλοιπου κειμένου μέσα από μία HTML σελίδα. Στην περίπτωσή μας όμως, αυτός ο μηχανισμός δεν είναι αρκετός. Το σύστημά μας θα πρέπει να υλοποιεί έναν έξυπνο αλγόριθμο ο οποίος θα είναι σε θέση να ξεχωρίσει το επιθυμητό κείμενο από κείμενο που μπορεί να αφορά το navigation menu ή κάποιες διαφημίσεις. Με απλά λόγια, ο μηχανισμός μας θα πρέπει να είναι φτιαγμένος με τέτοιον τρόπο ώστε να ανακτάται μόνον ο τίτλος και το κείμενο του άρθρου που αφορά κάποια είδηση. Κάθε άλλο κείμενο στη σελίδα είναι μη επιθυμητό και άρα ο μηχανισμός θα πρέπει να το απορρίπτει.

Τέτοιοι μηχανισμοί κατασκευάζονται σε πειραματικό επίπεδο και κυρίως για ερευνητικούς σκοπούς. Απλοϊκά προγράμματα που να μπορούν να απομονώσουν κομμάτι μίας HTML σελίδας και να ανακτήσουν την πληροφορία που βρίσκεται σε ένα συγκεκριμένο κομμάτι υπάρχουν, αλλά θα πρέπει να προσαρμοστούν σε κάθε διαφορετική ιστοσελίδα. Δεν είναι εφικτό να υπάρχει ένα γενικό σύστημα το οποίο να έχει τη δυνατότητα να αναλύσει τα σημεία που εντοπίζεται χρήσιμο κείμενο. Για το λόγο αυτό στηριζόμαστε στη θεωρία του web clipping σύμφωνα με την οποία είναι εφικτός ο διαχωρισμός περιοχών σε μία σελίδα και μάλιστα είναι εφικτό να δημιουργηθεί αλγόριθμος ο οποίος να εξάγει αυτόματα το χρήσιμο κείμενο από μία HTML σελίδα. Σε γενικές γραμμές οι μηχανισμοί αυτοί βασίζονται στο γεγονός πως η HTML σελίδα μπορεί να αναλυθεί σε δενδρική μορφή. Τα φύλλα του δένδρου αναπαριστούν το κείμενο που υπάρχει στη σελίδα με αποτέλεσμα να είναι εφικτό να εντοπιστούν άμεσα τα σημεία μέσα στο κείμενο που περιέχουν

κείμενο. Σε επόμενη φάση θα πρέπει να βρεθούν τα φύλλα τα οποία περιέχουν χρήσιμο κείμενο. Στην πιο απλή περίπτωση υπολογίζεται ο λόγος bytes κειμένου / bytes κώδικα + bytes κειμένου για κάθε κόμβο που έχει φύλλα. Με αυτό τον τρόπο επιτυγχάνεται το αυτονόητο. Σημεία που έχουν πολύ περισσότερο κείμενο απ' ότι κώδικα προφανώς και έχουν χρήσιμο κείμενο. Θέτοντας ένα αυστηρό όριο για το συγκεκριμένο λόγο έχουμε σαν αποτέλεσμα το να εντοπίσουμε τις θέσεις που έχουν αποκλειστικά και μόνο κείμενο. Ο αλγόριθμος που περιγράφηκε είναι απλός και αποτελεσματικός και συχνά χρησιμοποιείται ατόφιος σε όλα τα συστήματα εξαγωγής χρήσιμου κειμένου.

3.3 Προεπεξεργασία δεδομένων

Στη θεωρία, τα βασισμένα σε κείμενο χαρακτηριστικά ενός εγγράφου μπορούν να περιλαμβάνουν κάθε λέξη / φράση η οποία μπορεί να εμφανίζεται σε ένα δεδομένο σύνολο κειμένων. Όμως, επειδή κάτι τέτοιο είναι υπολογιστικά μη-ρεαλιστικό, χρειαζόμαστε κάποια μέθοδο προεπεξεργασίας κειμένων για την αναγνώριση των λέξεων - κλειδιών (κωδικολέξεων ή αλλιώς keywords) και φράσεων οι οποίες μπορεί να μας είναι χρήσιμες. Διάφορες τεχνικές έχουν προταθεί για την αναγνώριση των keywords ενός κειμένου όπως τα Hidden Markov Models [53], η Naive Bayes [104] και τα Support Vector Machines [80]· όμως όλες αυτές οι μέθοδοι τείνουν να κάνουν χρήση συγκεκριμένης γνώσης μετα-πληροφορίας για τη γλώσσα του κειμένου. Άλλες μέθοδοι χρησιμοποιούν στατιστικές πληροφορίες, όπως η συχνότητα μιας λέξης. Μια ευρέως γνωστή τεχνική είναι η TF-IDF (Term Frequency - Inverse Document Frequency), όπου TF είναι το πλήθος των εμφανίσεων ενός όρου σε ένα δεδομένο σύνολο κείμενων συγκρινόμενο με το πλήθος των κειμένων που περιέχουν το συγκεκριμένο όρο, και IDF είναι ένα μέτρο των συνολικών κειμένων σε μια συλλογή κειμένων, συγκρινόμενο με το συνολικό αριθμό κειμένων που περιέχουν μια δεδομένη λέξη [78]. Σχετικές τεχνικές, οι οποίες περιλαμβάνουν άλλες στατιστικές που πηγάζουν από το σύνολο των κειμένων, έχουν επίσης προταθεί τα πρόσφατα χρόνια· π. χ. κέρδος πληροφορίας [133], odds ratio [97], CORI [65], κλπ. Οι τεχνικές αυτές προσφέρουν μια βελτιωμένη προσέγγιση.

3.3.1 *stemming*

Στην ανάκτηση πληροφορίας, η σχέση μεταξύ ενός ερωτήματος χρήστη και ενός κειμένου καθορίζεται κυρίως από το πλήθος των όρων που έχουν κοινούς. Δυστυχώς, οι λέξεις έχουν πολλές μορφολογικές παραλλαγές οι οποίες δεν αναγνωρίζονται από αλγόριθμους που βασίζονται στο ταίριασμα όρων χωρίς να προηγηθεί κάποιας μορφής επεξεργασία φυσικής γλώσσας (Natural Language Processing). Στις περισσότερες των περιπτώσεων, αυτές οι παραλλαγές έχουν παρόμοιες εννοιολογικές ερμηνείες και μπορούν να αντιμετωπισθούν ως ισοδύναμες στα πλαίσια εφαρμογών ανάκτησης πληροφορίας (σε αντίθεση με τις γλωσσολογικές). Ως εκ' τούτου, ένα πλήθος αλγορίθμων κατάλληλων για τη διαδικασία του stemming έχουν αναπτυχθεί ώστε να περιορίσουν τις μορφολογικές παραλλαγές στην αρχική τους ρίζα.

Το πρόβλημα του stemming έχει προσεγγιστεί από μια μεγάλη ποικιλία μεθόδων που περιγράφονται στο [86] και περιλαμβάνουν αφαίρεση της κατάληξης, τμηματοποίηση λέξης και λεξιλογική μορφοποίηση. Δύο από τους διασημότερους αλγορίθμους, ο Lovins[88] και ο Porter[111], βασίζονται στην αφαίρεση της κατάληξης. Ο αλγόριθμος Lovins βρίσκει το μακρύτερο ταίριασμα από μια μεγάλη λίστα καταλήξεων, ενώ ο Porter [29] χρησιμοποιεί έναν επαναληπτικό αλγόριθμο με μικρότερο αριθμό καταλήξεων και μερικούς κανόνες. Ένας ακόμη αλγόριθμος, ο Paice/Husk [107], χρησιμοποιεί αποκλειστικά ένα σύνολο κανόνων ενώ ακολουθεί επαναληπτική προσέγγιση.

Στο [85] περιγράφονται τα προβλήματα που σχετίζονται με αυτές τις προσεγγίσεις. Οι περισσότεροι stemmers λειτουργούν χωρίς λεξικό και επομένως αγνοούν το νόημα των λέξεων, κάτι που οδηγεί σε ορισμένα λάθη κατά τη διαδικασία του stemming. Λέξεις διφορούμενες μειώνονται στην ίδια ρίζα και λέξεις με παρόμοιο νόημα δεν μειώνονται στην ίδια ρίζα. Για παράδειγμα, ο Porter stemmer μειώνει τις λέξεις general, generous, generation, generic στην ίδια ρίζα.

Παράλληλα, η έξοδος (stems) που παράγεται από τους αλγορίθμους, συνήθως δεν περιέχει πραγματικές λέξεις, κάτι που την κάνει δύσχρηστη για εργασίες που έχουν να κάνουν με ανάκτηση πληροφορίας. Διαδραστικές τεχνικές οι οποίες απαιτούν είσοδο από τον χρήστη απαιτούν από αυτόν την εργασία με stems και όχι πραγματικών λέξεων. Προβλήματα αυτού του τύπου αντιμετωπίζονται προσεγγίζοντας τη διαδικασία με μορφολογική ανάλυση.

Υπάρχει ένας μεγάλος αριθμός εργασιών που έχουν εξετάσει τον αντίκτυπο των stemming αλγορίθμων στην απόδοση της ανάκτησης πληροφορίας. Στο [63] δίνεται μια καλή περίληψη, αναφέροντας ότι τα συνδυασμένα αποτελέσματα των προηγούμενων μελετών καθιστούν ασαφές εάν η διαδικασία του stemming είναι χρήσιμη. Στις περιπτώσεις όπου το stemming είναι χρήσιμο τείνει να ασκήσει μόνο μικρή επίδραση στην απόδοση, και η επιλογή του stemmer μεταξύ των πιο κοινών παραλλαγών δεν είναι σημαντική. Εντούτοις, δεν υπάρχει κανένα στοιχείο ότι ένα λογικός stemmer μπορεί να βλάψει την απόδοση της ανάκτησης πληροφορίας.

Αντίθετα, μια πρόσφατη μελέτη [85] εντοπίζει μια αύξηση 15-35% στην απόδοση ανάκτησης όταν το stemming χρησιμοποιείται σε μερικές συλλογές (CACM και npl). Αναφέρεται ότι αυτές οι συλλογές έχουν και ερωτήματα και έγγραφα τα οποία είναι εξαιρετικά σύντομα. Για συλλογές με μεγαλύτερα κείμενα, οι stemming αλγόριθμοι χαρακτηρίζονται από μια σχετική αύξηση στην απόδοση της διαδικασίας ανάκτησης πληροφορίας.

3.4 Κατηγοριοποίηση πληροφορίας

Η αυτόματη *κατηγοριοποίηση* κειμένων είναι η διαδικασία ανάθεσης ετικετών κατηγορίας (προκαθορισμένων) σε νέα κείμενα που καταφθάνουν, στηριζόμενη στην πιθανότητα η οποία προτείνεται από τη βάση γνώσης που προϋπάρχει. Η διαδικασία έχει εγείρει ορισμένες προκλήσεις για τις στατιστικές μεθόδους που συνήθως χρησιμοποιούνται, και την αποτελεσματικότητά τους στην επίλυση πραγματικών προβλημάτων, τα οποία συχνά είναι πολλών διαστάσεων και έχουν μη σαφώς καθορισμένη κατανομή μεταξύ των κειμένων προς κατηγοριοποίηση. Η ανίχνευση του θέματος ενός κειμένου, για παράδειγμα, είναι η πιο κοινή εφαρμογή της κατηγοριοποίησης κειμένων. Ένας ολοένα και αυξανόμενος αριθμός μεθόδων αντιμετώπισης του προβλήματος προτείνονται, μεταξύ των οποίων μοντέλα παλινδρόμησης [58][132], κατηγοριοποίηση κοντινότερων γειτόνων [93][131], πιθανοτικές προσεγγίσεις με μεθόδους Bayes [126][87], επαγωγική εκμάθηση κανόνων [38][52], νευρωνικά δίκτυα [103], on-line εκμάθηση [52] και Support Vector Machines [76]. Παρότι η πλούσια βιβλιογραφία που υπάρχει πάνω στον τομέα της κατηγοριοποίησης κειμένων, ασφαλείς εκτιμήσεις και συγκρίσεις μεταξύ των μεθόδων είναι συνήθως δύσκολες.

Για να είναι δυνατή η παραγωγή μιας κατηγοριοποιημένης περίληψης, που θα ανταποκρίνεται στα ενδιαφέροντα του τελικού χρήστη, πρέπει να εντοπιστεί η κατηγορία του κειμένου. Λέξεις κλειδιά, οι οποίες είναι μοναδικές για κάποιο πεδίο (κατηγορία) αποτελούν πολύ καλές ενδείξεις για την κατηγορία του κειμένου [113]. Άλλες εναλλακτικές επιλογές, όπως συντακτικές και στατιστικές εκφράσεις έχουν επίσης χρησιμοποιηθεί [49][67][114]. Το βασικό θέμα της αναγνώρισης του θέματος με χρήση NLP έχει αναλυθεί διεξοδικά στο [75].

Άλλες επαναστατικές τεχνικές, όπως η χρήση κωδικών ελέγχου [43], η χρήση αιτιολογικών δικτύων έχουν προταθεί και αποτελούν ουσιαστικά μια τροποποιημένη έκδοση του Bayes αλγορίθμου του [119] που αποδίδουν καλά σε εργασίες κατηγοριοποίησης κειμένων. Καμία από τις προηγούμενες τεχνικές δεν αντιμετωπίζει τα σημασιολογικά θέματα.

3.4.1 Ταξινόμηση κειμένων

Δεδομένου ενός συνόλου πινάκων κειμένων $\{d_1, d_2, ..., d_n\}$ και των συσχετιζόμενων με αυτά ετικετών $c(d_i) \in \{c_1, c_2, ..., c_l\}$, η διαδικασία της ταξινόμησης αφορά στον καθορισμό της σωστής ετικέτας του νέου κειμένου d. Η ταξινόμηση κειμένων (text classification) έχει μελετηθεί σε μεγάλο βαθμό, ιδιαίτερα ύστερα από την εμφάνιση του διαδικτύου. Οι περισσότεροι αλγόριθμοι βασίζονται στο μοντέλο 'συνόλου λέξεων' του κειμένου [118]. Ένας απλός και συνάμα αποτελεσματικός αλγόριθμος είναι αυτός του Naive Bayes [96]. Για το πρόβλημα της ταξινόμησης κειμένων, διάφορες παραλλαγές του Naive Bayes έχουν χρησιμοποιηθεί αλλά έχει βρεθεί [95] ότι η παραλλαγή που βασίζεται στο πολυωνυμικό μοντέλο οδηγεί σε καλύτερα αποτελέσματα. Η μέθοδος των Support Vector Machines (SVMs) έχει επίσης χρησιμοποιηθεί με καλά αποτελέσματα [76][48]. Για ιεραρχικά δεδομένα κειμένων, όπως οι ιεραρχίες θεμάτων του Yahoo! [36] και το Open Directory Project [26], έχει μελετηθεί στα [84][50][57].

Για να αποφευχθούν οι πολλές διαστάσεις στην αναπαράσταση των κειμένων, πολλές μέθοδοι επιλογής χαρακτηριστικών έχουν προταθεί [133][84][50]. Επίσης συχνά επιζητείται η ιδιότητα της 'ισχυρής' ταξινόμησης όπου η κάθε λέξη του κειμένου μπορεί να αντιπροσωπευθεί από τη μοναδική ομάδα που ανήκει. Τέτοια

ιδιότητα αξιοποιείται στα [95][124]. Η επιλογή του μεγίστου πλήθους των λέξεων που θα απαρτίζουν ένα cluster είναι επίσης κάτι σημαντικό [128][115].

3.5 Αυτόματη εξαγωγή περίληψης

Παρουσιάζει ενδιαφέρον το γεγονός ότι πολλές διεργασίες ανάκτησης πληροφορίας, όπως η κατηγοριοποίηση κειμένου και η εξόρυξη πληροφορίας, μοιράζονται τους ίδιους στόχους και προβλήματα με την εξαγωγή περίληψης. Τα προβλήματα των συστημάτων ανάκτησης, λόγω του διλήμματος ακρίβειας - ανάκτησης, μπορούν να μειωθούν κάνοντας χρήση μιας αυτόματα εξαγόμενης περίληψης στοχευμένη στο προσωποποιημένο προφίλ (ενδιαφέροντα) του χρήστη.

Η έρευνα στον τομέα της αυτόματης περίληψης, θεωρούμενη ως εξαγωγή, αφαίρεση ή περίληψη χρήσιμου κειμένου, έχει μεγάλη ιστορία με αρχικό 'ξέσπασμά' τις προσπάθειες στη δεκαετία του 60 της πρωτοποριακής εργασίας του Luhn, ακολουθείται από τις δύο επόμενες δεκαετίες με σχετικά μικρή έρευνα στο θέμα, και κορυφώνεται τη δεκαετία του 90 και ως της μέρες μας με πολλές ερευνητικές προσπάθειες [108],[60],[90]. Σε κάθε περίπτωση, η δουλειά που έχει γίνει και που ουσιαστικά αφορά προτάσεις υλοποίησης κατατάσσονται σε δύο υποομάδες: εξαγωγή κειμένου και εξαγωγή γεγονότων. Στην εξαγωγή κειμένου, όπου 'αυτό που βλέπεις είναι αυτό που παίρνεις', μερικά τμήματα που υπάρχουν στο αρχικό κείμενο μεταφέρονται αυτούσια στην περίληψη του. Η εξαγωγή κειμένου είναι μια 'ανοιχτή' προσέγγιση στο πρόβλημα της περίληψης εφόσον δεν υπάρχει κάποια προηγούμενη υπόθεση για το τι είδους πληροφορία περιεχομένου είναι χρήσιμη. Το τι είναι σημαντικό για το πηγαίο κείμενο θεωρείται ως αξιοπρόσεκτο σε σχέση με κάποια γενικά, γλωσσολογικά, σημαντικά κριτήρια τα οποία εφαρμόζονται κατά τη διαδικασία εξαγωγής. Με την εξαγωγή γεγονότων αυτό που συμβαίνει είναι το αντίθετο: 'αυτό που ξέρεις είναι αυτό που παίρνεις', δηλαδή αυτό που έχεις ήδη αποφασίσει πως είναι το θέμα του περιεχομένου που αναζητάς στο πηγαίο κείμενο, αυτό είναι που τελικά παίρνεις στην περίληψη του. Αυτή είναι μια 'κλειστή' προσέγγιση, εννοώντας ότι το πηγαίο κείμενο δεν κάνει κάτι παραπάνω από το να παρέχει ένα στιγμιότυπο από κάποιες ήδη προκαθορισμένες απαιτήσεις. Η μέθοδος εξαγωγής κειμένου στοχεύει στο να κάνει το σημαντικό περιεχόμενο να 'αναδυθεί' μόνο του από κάθε κείμενο. Αντίθετα η μέθοδος εξαγωγής γεγονότων στοχεύει να βρει εμφανή στοιχεία σημαντικών ιδεών (γνωμών), ανεξαρτήτως της κατάστασης του κειμένου.

Οι τεχνικές προεπεξεργασίας που χαρακτηρίζουν τις δύο προαναφερόμενες μεθόδους εξαγωγής είναι πολύ διαφορετικές. Στην εξαγωγή κειμένου, η προεπεξεργασία στη ουσία συνενώνει τα στάδια ερμηνείας και μετασχηματισμού. Σημεία 'κλειδιά' του κειμένου, συνήθως ολόκληρες προτάσεις, αναγνωρίζονται από ένα μείγμα από στατιστικά, τοπικά και άλλα κριτήρια και επιλέγονται. Στη συνέχεια η παραγωγή της περίληψης είναι ουσιαστικά μια διαδικασία εξομάλυνσης των επιλεγμένων τμημάτων. Για παράδειγμα, διόρθωση αναφορών που περιέχονται σε επιλεγμένες προτάσεις και δεν αναφέρονται στην περίληψη. Θα μπορούσαμε να δούμε αυτή την στρατηγική εξαγωγής ως εξής: το πηγαίο κείμενο αντιμετωπίζεται χωρίς καμία ερμηνεία και η αναπαράστασή του τίθεται σε ένα στάδιο μετασχηματισμού το οποίο είναι στην ουσία εξαγωγικό. Η εξαγόμενη περίληψη είναι επομένως γλωσσολογικά 'κοντά' στο αρχικό κείμενο όσον αφορά την δομή της. Γενικά, με τις περιλήψεις που παράγονται με αυτόν τον τρόπο είναι σαν να έχουμε μια 'θολή εικόνα' για το αρχικό κείμενο. Οι επιλεγμένες προτάσεις συνήθως έχουν κάποια συσχέτιση μεταξύ τους αλλά και με το τμήμα του κειμένου που θα εκτιμούσαμε ως σημαντικό - το νόημά του. Όμως αυτή η μη εντελώς σαφής αναπαράσταση του αρχικού κειμένου γίνεται ακόμη πιο θολή δεδομένου ότι το εξαγόμενο κείμενο της περίληψης, παρότι εξομαλυμένο, δεν είναι συνήθως εντελώς κατανοητό. Αυτό αποτελεί και το σημαντικότερο πρόβλημα της μεθόδου αυτής.

Με την εξαγωγή γεγονότων, τα στάδια ερμηνείας και μετασχηματισμού επίσης ενώνονται. Η αρχική προεπεξεργασία κειμένου σχεδιάζεται ώστε να εντοπίζει και να επεξεργάζεται τα τμήματα του αρχικού κειμένου που σχετίζονται σε γενικές και προκαθορισμένες αρχές ή συσχετίσεις. Δεν υπάρχει ανεξάρτητη αναπαράσταση του πηγαίου κειμένου, μόνο άμεση εισαγωγή πηγαίου υλικού, αλλαγμένο λίγο έως πολύ σε σχέση με την αρχική του αναπαράσταση σύμφωνα με τις απαιτήσεις της κάθε ανεξάρτητης εφαρμογής.

Πιθανοτικά μοντέλα [82],[40] κατανομής των όρων στα κείμενα έχουν βρει χρησιμότητα στον τομέα της αυτόματης εξαγωγής περίληψης, το ίδιο και οι κλασικές TF-IDF (term frequency inverse document frequency) μέθοδοι [117] οι οποίες χρησιμοποιούνται στις περισσότερες εργασίες αυτόματης περίληψης κειμένων και παράγουν ένα ad-hoc σχήμα ζυγίσματος των λέξεων διότι δεν εξάγονται απ' ευθείας από κάποιο μαθηματικό μοντέλο κατανομής όρων ή σχετικότητας. Επιπλέον, κάποιες ερευνητικές εργασίες

[121] προσεγγίζουν το πρόβλημα με Poisson και αρνητικές διωνυμικές κατανομές ή με χρήση του k-mixture μοντέλου [120] το οποίο πλησιάζει το μοντέλο του αρνητικού διωνύμου αλλά είναι υπολογιστικά σημαντικά απλούστερο.

Στην πράξη παρατηρούνται σημαντικές παραλλαγές στις προαναφερόμενες μεθόδους προσέγγισης του προβλήματος που συχνά συσχετίζονται με τον επιθυμητό βαθμό μείωσης του μήκους εισόδου. Έτσι, για μικρές πηγές, η εξαγωγή μιας μοναδικής πρότασης μπορεί να φαντάζει σωστή (αν και επικίνδυνη) και αποφεύγει το πρόβλημα της συνοχής νοήματος των προτάσεων εξόδου (μιας και αυτή είναι μόνο μία). Παρόμοια, για τύπου μικρής εισόδου, μπορεί να είναι καταλληλότερη η επεξεργασία όλου του πραγματικού μήκους του κειμένου [134]. Από την άλλη μεριά, όπου η εξαγωγή περίληψης βασίζεται στην εξαγωγή γεγονότων από πολλές πηγές, μπορεί να απαιτούνται περισσότεροι μετασχηματισμοί των συνδυασμένων τους αναπαραστάσεων, όπως στο σύστημα POETIC [61], όπου η διαδικασία περίληψης είναι δυναμικά εξαρτώμενη από τα συμφραζόμενα. Είναι φανερό ότι χρειαζόμαστε α) περισσότερη αποτελεσματικότητα στην αυτοματοποιημένη περίληψη από ότι η εξαγωγή κειμένου μας προσφέρει και β) περισσότερη ευελιξία από ότι η εξαγωγή γεγονότων μας παρέχει.

Πέρα από τη διαδικασία εξαγωγής, είναι σημαντικός ο ρόλος της δομής του κειμένου αλλά και των συμφραζομένων στην εξαγωγή αποτελεσματικής περίληψης. Βελτιώσεις επομένως στη διαδικασία περίληψης θα περιλαμβάνουν μεθόδους σύλληψης της δομής αυτής στο αρχικό κείμενο και χρήση της κατά τη διαδικασία εξαγωγής των χρήσιμων τμημάτων του κειμένου. Παραδείγμα της προσπάθειας αυτής αποτελεί η Rhetorical Structure Theory[92]. Οι προσεγγίσεις που εφαρμόζονται συνήθως έχουν να κάνουν με το είδος της πληροφορίας, γλωσσολογικά, επικοινωνιακά πεδία ενδιαφέροντος που καθορίζουν τη δομή, με το είδος της δομής και τις συσχετίσεις μεταξύ δομών διαφόρων ή του ίδιου κειμένου.

Συνοπτικά θα λέγαμε ότι διακρίνουμε δύο κύριους τρόπους εξαγωγής της περίληψης του αρχικού κειμένου. Ο πρώτος είναι οι ευρετικές μέθοδοι, που βασίζονται κυρίως στον τρόπο σκέψης και εργασίας του ανθρώπου. Πολλές από αυτές, αξιοποιούν την όποια οργάνωση του εγγράφου. Έτσι, προτάσεις που βρίσκονται στις αρχικές και τις τελικές παραγράφους του κειμένου είναι πολύ πιθανό να περιέχονται στην τελική περίληψη. Ο δεύτερος τρόπος, αποτελείται από μεθόδους που βασίζονται στην αναγνώριση λέξεων κλειδιών, φράσεων και ομάδων λέξεων. Το έγγραφο αναλύεται με την χρήση στατιστικών ή/και γλωσσολογικών τεχνικών, για να βρεθούν τα στοιχεία εκείνα που αναπαριστούν το περιεχόμενο του εγγράφου. Αφού ολοκληρωθεί η διαδικασία της περίληψης, ορισμένοι περιλήπτες επιτελούν κάποια περιορισμένη μετα-επεξεργασία ομαλοποίησης των προτάσεων της περίληψης. Δημιουργούν μία λίστα προτάσεων, σε μία προσπάθεια να δοθεί συνέπεια και ευφράδεια στην περίληψη. Γενικά, απομακρύνουν τα ακατάλληλα συνδετικά λέξεων και φράσεων, και εξακριβώνουν σε ποιόν αναφέρονται οι αντωνυμίες του κειμένου ώστε η τελική περίληψη να έχει μια συνοχή.

3.5.1 Συστήματα περίληψης βασισμένα στη γνώση

Από την γέννηση τους, η ανάπτυξη των συστημάτων αντίληψης κειμένων ήταν άρρηκτα συνδεδεμένη με το πεδίο της αναπαράστασης γνώσης και των μεθόδων λογικής [122]. Αυτή η στενή σχέση αιτιολογήθηκε από την παρατήρηση ότι για να έχουμε μια επαρκή κατανόηση του κειμένου απαιτείται γραμματική γνώση σχετικά με τη συγκεκριμένη γλώσσα του κειμένου, αλλά και ενσωμάτωση προηγούμενης γνώσης με την οποία πραγματεύεται το κείμενο. Έτσι, οι συμπερασματικές δυνατότητες των γλωσσών αναπαράστασης γνώσης θεωρούνται πολύ σημαντικές για συστήματα που θα κατανοούν κείμενα. Βασισμένα σε αυτού του είδους την αντίληψη, μια σειρά από συστήματα εξαγωγής περίληψης, βασισμένα στην αναπαράσταση γνώσης, αναπτύχθηκαν (Schankian-type Conceptual Dependency reprecentations). Τα συστήματα αυτά αποτέλεσαν την πρώτη γενιά συστημάτων δημιουργίας αυτοματοποιημένης περίληψης βασιζόμενα στη γνώση.

Ακολούθησε μια δεύτερη γενιά συστημάτων η οποία υιοθέτησε μια πιο 'ώριμη' προσέγγιση αναπαράστασης γνώσης, βασιζόμενη στην ήδη υπάρχουσα μεθοδολογία υβριδικών, βασισμένων σε κατηγοριοποίηση, γλωσσών αναπαράστασης [130]. Αυτές οι αρχές χρησιμοποιήθηκαν σε συστήματα περίληψης όπως τα: SUSY [66], SCISOR [74] και TOPIC [70]. Αλλά ακόμη και αυτού του είδους τα συστήματα αδυνατούσαν να εξάγουν αποτελεσματικά αξιόλογες μεταφράσεις.

3.5.2 Αναγνώριση Θεμάτων

Το θέμα της *αναγνώρισης θεμάτων (Topic Identification)*, αναφέρεται στην διαδικασία της έρευνας σε έγγραφα κειμένου, για την ανακάλυψη συγκεκριμένων δομών. Σύμφωνα με τους Mather A. Laura και Note Jarrod [94], μία ολοκληρωμένη εφαρμογή, που θα αφορά το θέμα της εύρεσης θεμάτων, θα πρέπει να έχει τη δυνατότητα επεξεργασίας εγγράφων κειμένου, με σκοπό την ανακάλυψη κανόνων και αλγορίθμων, που θα αναγνωρίζουν εγκυκλοπαιδική δομή και εγκυκλοπαιδικά θέματα. Αν αναγνωριστούν συγκεκριμένα θέματα σε έγγραφα κειμένου, τότε αυτά μπορούν να αξιοποιηθούν κατάλληλα και να ενσωματωθούν σε κάποια εγκυκλοπαίδεια. Με αυτό τον τρόπο η εγκυκλοπαίδεια θα είναι ενημερωμένη και η εταιρεία που διαχειρίζεται μία τέτοια εφαρμογή, θα έχει σίγουρα ένα ανταγωνιστικό πλεονέκτημα έναντι των υπολοίπων. Για την υλοποίηση αυτή, απαιτείται η χρησιμοποίηση της επεξεργασίας φυσικής γλώσσας (Natural Language Processing), η ανάκτηση πληροφορίας (Information Retrieval) και η *υπολογιστική γλωσσολογία (Computational Linguistics)*. Αρχικά απαιτείται η αναγνώριση περιοχών δευτερεύουσας σημασίας (Subtopic Regions) μέσα στο κείμενο, και στην συνέχεια η εύρεση των θεμάτων που σχετίζονται με τις περιοχές αυτές. Για τους σκοπούς αυτούς, αναγνωρίζονται οι φράσεις των ουσιαστικών, τα όρια των προτάσεων και των παραγράφων του κειμένου *(Tokenization)* . Στην συνέχεια, απομακρύνονται όλες οι συχνές λέξεις *(stopwords)*, μετατρέπεται κάθε λέξη στον ενικό αριθμό και υπολογίζεται η ρίζα της κάθε λέξης. Ακολούθως, ανακαλύπτονται οι περιοχές δευτερεύουσας σημασίας (Subtopic Regions) και προστίθενται ετικέτες στο κείμενο που έχει επεξεργαστεί μέχρι τώρα, για την αναγνώριση των ορίων του κάθε επιθέματος. Τέλος, αναγνωρίζονται τα προεξέχοντα και τα δευτερεύουσας σημασίας θέματα του εγγράφου (Topics, Subtopics). Αφού βρεθούν οι περιοχές δευτερεύουσας σημασίας, υπολογίζεται η βαθμολογία του κάθε θέματος, η οποία θα υποδείξει την υπεροχή του αντίστοιχου θέματος στην αντίστοιχη περιοχή.

3.5.3 Περίληψη κειμένου βασισμένη στο χρόνο

Παρότι είναι λίγη σχετικά η έρευνα στο συγκεκριμένο τομέα, ορισμένοι ερευνητές έχουν ασχοληθεί με το πως είναι δυνατή η εξαγωγή προσωρινών εκφράσεων από ένα κείμενο, αναζητώντας και κανονικοποιώντας αναφορές σε ημερομηνίες, χρόνο και παρερχόμενο χρόνο [91]. Η δουλειά αυτή είναι σημαντική για την ανάλυση του περιεχομένου του κειμένου αλλά όχι για αυτή καθ' αυτή την περίληψή του. Το 1999, το Novelty Detection workshop στο Πανεπιστήμιο του Johns Hopkins εισήγαγε το New Information Detection - NID, έργο του οποίου ήταν η καταγραφή της 'νέας' πληροφορίας σε ένα θέμα επισημαίνοντας την πρώτη πρόταση που την περιείχε [37]. Προβλήματα σχετικά με τον επιτυχή καθορισμό της έννοιας 'νέο' εμπόδισαν το σύστημα αυτό ώστε να επιτύχει. Η έρευνα αυτή σχετίζεται και με τον τομέα του automatic timeline construction [125] που επικεντρώνεται στην εξαγωγή ασυνήθιστων λέξεων και φράσεων από μία συνεχή ροή νέων και στην περαιτέρω ομαδοποίηση των συστατικών αυτών ώστε να απομονωθούν θέματα μέσα σε ένα νέο.

3.5.4 Αξιολόγηση της περίληψης κειμένου

Μια περίληψη κειμένου είναι γενικά δύσκολο να *αξιολογηθεί*, κυρίως λόγω των υποκειμενικών κριτηρίων που τίθενται. Ανακατανομή τμημάτων του κειμένου, προτάσεων, παράληψη προφανώς ασήμαντων φράσεων, κ.ο.κ. όλα αυτά καταλήγουν σε μια μεγάλη ποικιλία 'καλών' περιλήψεων. Πώς καταλήγουμε όμως στην καλύτερη περίληψη και πως μπορούμε να πούμε πως αυτή που παράγει ο μηχανισμός μας προσεγγίζει τη βέλτιστη;

Υπάρχουν γενικότερα οι εξής μέθοδοι που χρησιμοποιούνται για την αξιολόγηση μια εξαγόμενης περίληψης:

- Χρήση αρκετών πρωτοτύπων παραδειγμάτων από τεχνικές περίληψης κειμένου για τις οποίες γνωρίζουμε την απόδοσή τους
- Συμμετοχή ανθρώπων [46][106] με την ανάγνωση των περιλήψεων και την βαθμολόγησή τους με κριτήριο το πόσο αντιπροσωπευτική θεωρείται σε σχέση με το αρχικό κείμενο
- Θεωρούμε ότι η περίληψη του κειμένου είναι ένα υποσύνολο του κειμένου και ελέγχουμε εάν μπορεί να αντιπροσωπεύσει επαρκώς το αρχικό κείμενο σε θέματα όπως: είναι δυνατό να κατηγοριοποιηθεί

το κείμενο με βάση την περίληψή του ή να εντοπιστεί εάν ανταποκρίνεται στις προτιμήσεις του χρήστη χωρίς να εξεταστεί το αρχικό κείμενο [62][105]; Μπορεί ένας χρήστης να εμπεδώσει σωστά το κείμενο έχοντας διαβάσει μόνο την περίληψή του και απαντώντας σε tests [102]; Μπορεί ο χρήστης να αντιστοιχίσει σωστές λέξεις - κλειδιά σε μια περίληψη [116];

- Συγκρίνουμε την ομοιότητα μεταξύ προτάσεων επιλεγμένων από ανθρώπους, ως αντιπροσωπευτικές για το κείμενο, και των προτάσεων που προέκυψαν από την αυτοματοποιημένη περίληψη [69][112], ή συγκρίνουμε το βαθμό αντιπροσωπευτικότητας που δίνουν οι χρήστες σε μια πρόταση σε σχέση με αυτόν που δίνει ο μηχανισμός [56]. Οι τεχνικές αυτού του είδους αναφέρονται συνήθως και ως corpus-based.

3.5.5 Παραδείγματα συστημάτων

Ακολουθούν ορισμένα σημαντικά συστήματα αυτόματης εξαγωγής περίληψης που χρίζουν αναφοράς.

Copernic Summarizer

Πρόκειται για ένα εμπορικό προϊόν το οποίο πραγματοποιεί αυτόματη εξαγωγή περίληψης στα Αγγλικά, Γαλλικά και Γερμανικά. Χρησιμοποιείται για να παράγει περιλήψεις κειμένων και δικτυακών τόπων προσφέροντας με αυτό τον τρόπο μία γενική εικόνα των εγγράφων προτού ο χρήστης τα διαβάσει ολόκληρα.

Χρησιμοποιώντας πολύπλοκους στατιστικούς αλγορίθμους και γλωσσική ανάλυση, εντοπίζει τις πιο καίριες εκφράσεις του κειμένου και εξάγει τις πιο σημαντικές προτάσεις τόσο σε ένα δικτυακό τόπο όσο και σε ένα κείμενο. Ενώνοντας αυτές τις προτάσεις παράγεται η περίληψη του κειμένου.

Ως εμπορικό πρόγραμμα, δεν είναι εφικτή η αναλυτική προσέγγιση των τρόπων με τους οποίους πραγματοποιείται η εξαγωγή περίληψης.

MS Word Summarizer

Η εφαρμογή MS Word στις πιο πρόσφατες εκδόσεις της περιέχει ένα μηχανισμό αυτόματης εξαγωγής περίληψης κειμένων το οποίο απαρτίζεται από προτάσεις του κειμένου που απομονώνονται. Αναλυτικές πληροφορίες για τις μεθόδους που χρησιμοποιούνται για την εξαγωγή περίληψης δεν υπάρχουν, ωστόσο τα αποτελέσματα του μηχανισμού δεν είναι καθόλου ικανοποιητικά συγκριτικά με αλγορίθμους και μηχανισμούς που υπάρχουν.

MEAD Summarizer

Ο MEAD περιλήπτης είναι μια ελεύθερα διαθέσιμη σειρά εργαλειών για πολυ-γλωσσική περίληψη και αξιολόγηση. Χρησιμοποιεί πολλούς αλγορίθμους περίληψης π. χ. keyword-based, TF*IDF. Είναι γραμμένος σε γλώσσα Perl και πρόκειται ίσως για τον πιο ολοκληρωμένο μηχανισμό αυτόματης εξαγωγής περίληψης.

SUMMARIST

Ο Summarist είναι ένας μηχανισμός ο οποίος πραγματοποιεί αυτόματη εξαγωγή περίληψης κειμένων. Πρόκειται για ένα σύστημα το οποίο βασίζεται σε οντολογίες προκειμένου να αποκτήσει γνώση επί των λέξεων και χρησιμοποιεί αμιγώς NLP (Natural Language Processing). Η βασική συνάρτηση στην οποία στηρίζεται είναι:

$$\underline{\text{Κατηγοριοποίηση} = \text{Εντοπισμός τίτλου} + \text{μετάφραση} + \text{παραγωγή}}$$

Για κάθε βήμα από τα παραπάνω το σύστημα εφαρμόζει τις ακόλουθες τεχνικές:

- Εντοπισμός Τίτλου. Με γενίκευση των τεχνικών ανάκτησης πληροφορίας και προσθέτοντας τεχνικές εντοπισμού τίτλου, χρησιμοποιείται ο μηχανισμός SENSUS αλλά και λεξικά, ο μηχανισμός πραγματοποιεί εντοπισμό σεναρίων μέσα στο κείμενο. Επιτρέπει πολυγλωσσική ανάλυση και πιο συγκεκριμένα

οι γλώσσες στις οποίες πραγματοποιείται ο εντοπισμός είναι: Αγγλικά, Ισπανικά, Ιαπωνικά, Ινδονησιανά και Αραβικά.

- Μετάφραση. Το κομμάτι αυτό του μηχανισμού δεν κάνει τη μετάφραση των κειμένων αλλά χρησιμοποιεί τεχνικές στατιστικής ανάλυσης από την Ανάκτηση Πληροφορίας αλλά και LSA (Latent Semantic Analysis) όπως και λεξικά για να πραγματοποιήσει διασύνδεση των τίτλων και των σεναρίων που έχουν εντοπιστεί σε ένα κείμενο προκειμένου να εντοπιστεί το 'νόημα' του κειμένου.

- Δημιουργία. Ο μηχανισμός χρησιμοποιεί τρια διαφορετικά συστήματα για τη δημιουργία της αυτόματης περίληψης: μία λίστα λέξεων-κλειδιών, ένα μηχανισμό δημιουργίας φράσεων και ένα μηχανισμό δημιουργίας προτάσεων από λέξεις κλειδιά και φράσεις. Οι τρεις μηχανισμού λειτουργού σειριακά με τον τρόπο που αναφέρονται προκειμένου να δημιουργήσουν το επιθυμητό αποτέλεσμα.

3.6 Προσωποποίηση στο χρήστη

Σύμφωνα με τον Mobasher [98], η προσωποποίηση στο διαδίκτυο μπορεί να περιγραφεί σαν κάθε ενέργεια που σαν σκοπό έχει να κάνει τη Διαδικτυακή εμπειρία ενός χρήστη να είναι βάσει των αναγκών που έχει κάθε χρήστης. Σε γενικές γραμμές αυτό σημαίνει αλλαγή της παρουσίασης των δεδομένων ενός Δικτυακού τόπου προς το χρήστη σύμφωνα με τις εκάστοτε ρητές και εννοούμενες επιλογές του χρήστη. Αυτό είναι σχετικά εύκολο όταν αναφέρομαστε σε ένα και μόνον δικτυακό τόπο. Ο χρήστης καλείται να δηλώσει ρητά τις προτιμήσεις του ενώ παράλληλα το σύστημα 'μαθαίνει' τις προτιμήσεις του χρήστη. Αυτό συναντάται σε πολλούς δικτυακούς τόπους.

Ο έλεγχος της δραστηριότητας του χρήστη σε πολλαπλούς δικτυακούς τόπους και ο εντοπισμός των πραγματικών αναγκών του και επιλογών είναι μία μεγάλη πρόκληση. Αυτό συνεπάγεται πως τη στιγμή που ένας χρήστης επισκέπτεται ένα δικτυακό τόπο, υπάρχει ήδη ένα προφίλ του και το σύστημα είναι άμεσα σε θέση να προσαρμοστεί στις ανάγκες του συγκεκριμένου χρήστη. Πολλές προσεγγίσεις πάνω στο συγκεκριμένο θέμα έχουν δοκιμαστεί: Single Sign On συστήματα [27, 8], προσωποποίηση στη μεριά του χρήστη [77] και βέβαια όλα τα συστήματα spyware και ad trackers. Πολλά από αυτά τα συστήματα παρουσιάζουν προβλήματα με τη νομοθεσία καθώς προσβάλλουν την ιδιωτικότητα του χρήστη ενώ τα συστήματα που εφαρμόζουν την προσωποποίηση στη μεριά του χρήστη έχουν χαμηλή αποδοτικότητα.

Μία σειρά από πρωτοβουλίες στην W3C έχουν σαν σκοπό την καθολική προσωποποίηση. Το OPS (Open Profiling Standard) [71] είναι ένα προτεινόμενο W3C standard το οποίο έχει υποβληθεί από τις εταιρίες Netscape, Verisign και Firefly από το 1997. Παρουσιάζει ένα σχήμα τυποποίησης και ένα πρωτόκολλο ανταλλαγής δεδομένων που αφορούν το προφίλ ενός χρήστη, όπως για παράδειγμα το όνομα, τη διεύθυνση και τον ταχυδρομικό κώδικα. Ωστόσο, δεν τέθηκε ποτέ σε χρήση. Η ιδέα ανταλλαγής πληροφορίας είναι πολύ χρήσιμη, όμως πολλοί χρήστες δε θα επιθυμούσαν τη δημοσιοποίηση τέτοιων στοιχείων. Για την προσωποποίηση θα ήταν χρησιμότερο να διαμοιράζονται πληροφορίες που αφορούν την περιαγωγή ενός χρήστη στους δικτυακούς τόπους.

Το PIDL (Personalized Information Description Language) [83] είναι ένα πρωτόκολλο που υποβλήθηκε στην W3C από την εταιρία NEC το 1999. Πρόκειται για έναν τρόπο δόμησης εγγράφου που περιέχει στοιχεία για τις προτιμήσεις ενός χρήστη κατά τη διάρκεια που βρίσκεται σε διάφορους δικτυακούς τόπους. Είναι προφανές πως κάτι τέτοιο έρχεται ενάντια στα στοιχεία ιδιωτικότητας του χρήστη που έχουμε ήδη αναφέρει. Είχε προταθεί αρχικά για χρήση σε μυλτιαστ, μία τεχνολογία που τελικά δεν αναπτύχθηκε όσο αναμενόταν.

Το CC/PP (Composite Capabilities/Preference Profiles) [81] είναι ένα W3C στάνταρ που προτάθηκε το 1999 και βρίσκεται μέχρι και σήμερα σε χρήση. Επιτρέπει σε κινητούς χρήστες να εκφράσουν τις προτιμήσεις ενός χρήστη σε έναν κεντρικοποιημένο εξυπηρετητή. Παρά το γεγονός ότι οι κινητές τεχνολογίες έχουν πολλούς περιορισμούς στην ανταλλαγή δεδομένων, αυτή η αρχιτεκτονική θα μπορούσε να αποτελέσει τη βάση για ένα σύστημα διαμοιρασμού των προτιμήσεων ενός χρήστη.

Το P3P (Platform for Privacy Preferences) [55] έρχεται σε αντίθεση με κάθε σύστημα προσωποποίησης που βασίζεται στο διαμοιρασμό των στοιχείων ενός χρήστη μεταξύ δικτυακών τόπων. Αυτή η σύσταση της W3C που έγινε το 2002 έχει σχεδιαστεί ώστε να επιτρέπει στους χρήστες να ελέγχουν τα προσωπικά τους δεδομένα που θα παρουσιάζονται στους διάφορους δικτυακούς τόπους που επισκέπτεται.

Κανένα από τα παραπάνω δεν επιτρέπει την προσωποποίηση σε πολλαπλούς δικτυακούς τόπους. Αν αναλογιστούμε τα εμπορικά συστήματα θα δούμε πως πρόκειται για ένα σημαντικό κομμάτι τους, κυρίως όσον

αφορά θέματα μάρκετινγκ. Οι εταιρίες επιθυμούν να γνωρίζουν τις ανάγκες των 'πελατών' τους πρωτού αυτοί επισκευθούν το 'κατάστημά' τους. Έτσι, πολλοί δικτυακοί τόποι, όπως για παράδειγμα η προσωποποίηση και οι συστάσεις που παρουσιάζονται στο δικτυακό τόπο του Amazon.com [1] το εφαρμόζουν σε ατομικό επίπεδο. Από τις πρώτες κιόλας σελίδες που επισκέπτεται ο χρήστης διαμορφώνεται ένα προφίλ του προκειμένου ο δικτυακός τόπος να προσαρμόζεται σιγά - σιγά στις ανάγκες του.

Η μελέτη του θέματος που αφορά τις επιλογές ενός χρήστη καθώς και τη συμπεριφοράς αυτού κατά την επίσκεψη πολλών διαφορετικών δικτυακών τόπων έχει πραγματοποιηθεί από πολλές εταιρίες και έχουν γίνει πολλές προτάσεις. Αν εξαιρέσουμε τις προσπάθειες στις οποίες ανακύπτουν ηθικά αλλά και νομικά ζητήματα παραβίασης της ιδιωτικότητας καταλήγουμε αποκλειστικά στα συστήματα SSO (Single Sign On) όπως είναι το Microsoft Passport [27] και το Liberty Alliance [99]. Αυτά παρέχουν μία ενιαία βάση δεδομένων που περιέχει τα προσωπικά στοιχεία και τις επιλογές του. Οι χρήστες προσθέτουν από μόνοι τους στοιχεία στη βάση δεδομένων στα οποία έχουν ελεύθερη πρόσβαση εταιρίες που είναι συμβεβλημένες με τα εκάστοτε SSO συστήματα.

Βασικό πρόβλημα αυτής της προσέγγισης είναι η εξασφάλιση της ασφάλειας του συστήματος καθότι ο χρήστης μπορεί να αποθηκεύει ευαίσθητα δεδομένα. Το συγκεκριμένο θέμα τονίζεται ακόμα και στα προϊόντα των εταιριών (για παράδειγμα η Sun το τονίζει ιδιαίτερα στο πρόγραμμα Liberty. Πως θα εμπιστευτεί ένας χρήστης το πρόγραμμα το οποίο του τονίζει ιδιαίτερα πως δεν είναι ασφαλές· Τα νεότερα SSO συστήματα όπως το Liberty Alliance και το SXIP έχουν δώσει ιδιαίτερη προσοχή στο συγκεκριμένα θέμα προκειμένου να βελτιωθούν. Μάλιστα το SIXP επιτρέπει σε ένα χρήστη να διαθέτει πολλαπλά προφίλ ανάλογα με το μέγεθος των δεδομένων που επιθυμεί να είναι ορατά σε διάφορους δικτυακούς τόπους ορίζοντας με αυτό τον τρόπο αυτόνομα το επίπεδο ασφάλειας. Παράλληλα είναι ένα σύστημα ανοιχτού κώδικα προκειμένου οι χρήστες να μπορούν να δουν επακριβώς τι στοιχεία τους διαμοιράζονται και με ποιον τρόπο. Αυτό βέβαια δεν ξεπερνά τα προβλήματα που παρουσιάζονται. Οι χρήστες πρέπει να αποφασίσουν αν οι εταιρίες στις οποίες θα εμπιστευτούν τα προσωπικά τους δεδομένα είναι έμπιστες ή όχι. Αυτό συνεπάγεται και την αποτυχία τετοιων συστημάτων με χαρακτηριστικό παράδειγμα το σύστημα Passport σαν τεχνολογία καθότι οι χρήστες δεν έχουν κάποια ιδιαίτερη προτίμηση στα SSO συστήματα. Παράλληλα, όπως αναφέρει και ο Gartner [10], 'όσο οι χρήστες δε δείχνουν να αποδέχονται τέτοια συστήματα οι εταιρίες δεν πρόκειται να κάνουν απολύτως καμία επένδυση'.

Υπάρχουν βέβαια και συστήματα τα οποία δεν απαιτούν την εισαγωγή στοιχείων από το χρήστη αλλά χρησιμοποιούν μεταδεδομένα που υπάρχουν από τα ίχνη που αφήνει ένας χρήστης καθώς πραγματοποιεί περιαγωγή σε σελίδες του διαδικτύου. Το WAWA (Wisconsin Adaptive Web Assistant) [123] είναι ένα σύστημα το οποίο προσπαθεί να εντοπίσει τις σελίδες που μπορεί να αφορούν κάποιο χρήστη ανάλογα με το history που εντοπίζει στο φυλλομετρητή. Αντίστοιχα το Syskill and Webert [109] είναι ένα πρόγραμμα το οποίο μαθαίνει να βαθμολογεί τις σελίδες που επισκέπτεται ο χρήστης και αποφασίζει ποιες είναι οι σελίδες που πιθανόν ενδιαφέρουν το χρήστη. Το σύστημα αυτό χρησιμοποιεί το προφίλ χρήστη που το ίδιο κατασκευάζει και προτείνει στο χρήστη συνδέσμους που ενδεχόμενα τον ενδιαφέρουν το χρήστη ή πραγματοποιεί ερωτήματα σε μηχανές αναζήτησης με λέξεις κλειδιά από το διαμορφωμένο προφίλ χρήστη. Ο Chan [51] περιγράφει ένα παραπλήσιο σύστημα το οποίο περιέχει δύο στοιχεία: το Web Access Graph (WAG) και τον Page Interest Estimator (PIE). Το WAG εντοπίζει ίχνη σε ιστοσελίδες που μπορεί να αφορούν το χρήστη και το PIE 'μαθαίνει' τον τρόπο με τον οποίο επισκέπτεται ένας χρήστης μία σελίδα βάσει των επιλογών που κάνει.

Οι Widyantoro, Ioerger Yen [129] ανέπτυξαν ένα σύστημα το οποίο βασίζεται σε έναν τριπλό περιγραφέα προκειμένου να καταγράφουν τη δυναμική ενός χρήστη απέναντι στο διαδίκτυο. Το μοντέλο αυτό διατηρεί μία μία περιγραφή για κάθε ίχνος που αφήνει ο χρήστης στο διαδίκτυο σε ένα μεγάλο βάθος χρόνου και το συνδυάζει με δεδομένα που αποθηκεύονται προσωρινά προκειμένου να κάνει προβλέψεις για τις ιστοσελίδες που μπορεί να αφορούν το χρήστη.

Οι Goecks Shavlik [68] προτείνουν ένα σύστημα που 'μαθαίνει' τα ενδιαφέροντα του χρήστη ελέγχοντας περισσότερα στοιχεία που αφορούν τις σελίδες που επισκέπτεται. Παρατηρούν για παράδειγμα τις κινήσεις που κάνει ο χρήστης με το ποντίκι εκτός από την απλή διαδικασία ελέγχου των σελίδων που επισκέπτεται ο χρήστης.

ΚΕΦΑΛΑΙΟ 4

Αρχιτεκτονική και χαρακτηριστικά του Συστήματος

> Strength does not come from physical capacity. It comes from an indomitable will.
>
> *Mahatma Gandhi, Indian political and spiritual leader*

Στο τρέχον κεφάλαιο, περιγράφεται η αρχιτεκτονική του συστήματος που αναπτύχθηκε και οι στόχοι πάνω στους οποίους βασίζεται. Γίνεται παρουσίαση όλων των στοιχείων από τα οποία αποτελείται το σύστημα (υποσυστήματα), ενώ παράλληλα παρουσιάζεται ο τρόπος με τον οποίο γίνεται η εσωτερική διασύνδεση όλων των υποσυστημάτων καθώς και ο τρόπος με τον οποίο το σύστημα μπορεί να αξιοποιηθεί για χρήση σε συσκευές μικρού μεγέθους.

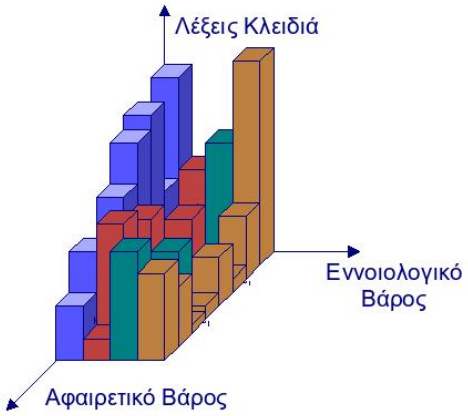
4.1 Χαρακτηριστικά του συστήματος

Η αρχιτεκτονική του συστήματος είναι αρκετά πολύπλοκη και περιλαμβάνει αρκετά υποσυστήματα που επιτελούν τις επιμέρους λειτουργίες. Αποτελεί επομένως έναν τμηματοποιημένο μηχανισμό, κάθε κομμάτι του οποίου μπορεί να λειτουργήσει και αυτόνομα. Η επιθυμητή αυτή ιδιότητα επιτυγχάνεται με τη χρήση γενικά αποδεκτών προτύπων για τη διασύνδεση των διαφόρων υποσυστημάτων. Κάθε υποσύστημα δέχεται είσοδο σε μορφή XML και παράγει έξοδο στην ίδια μορφή κάνοντας έτσι εύκολη τη διαχείριση τους. Είναι επομένως εύκολο να αντικατασταθεί ένα τμήμα (module) του συστήματος από ένα νεότερο ή καλύτερο, μπορούμε π. χ. σε μελλοντική προσπάθεια να αντικαταστήσουμε το μηχανισμό της περίληψης με έναν νεότερο και αποτελεσματικότερο μηχανισμό χωρίς να χρειαστεί να πειράξουμε τα υπόλοιπα υποσυστήματα.

4.1.1 Στόχοι του συστήματος

Το σύστημα έχει σχεδιαστεί προκειμένου να παρέχει ως έξοδο, στο χρήστη ή σε άλλα συστήματα, ποιοτική πληροφορία. Όπως έχει ήδη αναφερθεί στα προηγούμενα κεφάλαια, η πληροφορία του παγκοσμίου ιστού είναι σχεδόν χαοτική με αποτέλεσμα οι χρήστες να μην είναι εφικτό να προσεγγίσουν πληροφορία που τους είναι χρήσιμη και επιθυμητή. Σκοπός του συστήματός μας είναι να δημιουργήσουμε την κατάλληλη υποδομή ούτως ώστε να πραγματοποιείται φιλτράρισμα στην πληροφορία που κινείται στο διαδίκτυο και να αξιοποιείται προτού φτάσει στο χρήστη. Αυτό που θέλουμε να επιτύχουμε μέσα από αυτή τη διαδικασία είναι να αξιολογήσουμε κατά κάποιο τρόπο την πληροφορία που διακινείται στον παγκόσμιο ιστό και να βρούμε τα κανάλια επικοινωνίας που παρέχουν ποιοτική πληροφορία, ικανή να παρέχει σαφή και σφαιρική ενημέρωση στον χρήστη. Συχνά θα έχει παρατηρηθεί να μιλούμε για ενημέρωση και ποιότητα στην ενημέρωση που

παρέχει το διαδίκτυο. Αυτό συμβαίνει διότι το σύστημά μας θα αντλεί και θα επεξεργάζεται περιεχόμενο που εντοπίζεται σε ειδησεογραφικούς δικτυακούς τόπους. Το περιεχόμενό τους θα παραλαμβάνεται καθημερινά, θα φιλτράρεται, θα αναλύεται, θα κατηγοριοποιείται και θα περιλήπτεται. Έτσι στο τέλος θα έχουμε στα χέρια μας μια κατηγοριοποιημένη περίληψη της πληροφορίας, έτοιμη να παρουσιαστεί στο χρήστη ανάλογα με τις προτιμήσεις του, ή να μεταδοθεί σε άλλα ενδιάμεσα συστήματα. Παράλληλα, το σύστημα δε θα αφήνει τον τελικό χρήστη έξω από τη διαδικασία. Η κατηγοριοποίηση που θα πραγματοποιείται θα είναι σε δύο επίπεδα, τόσο ένα οριζόντιο που θα αφορά το γλωσσολογικό και εννοιολογικό κομμάτι, όσο και το κομμάτι της σχετικότητας και της ανάλυσης. Αυτό είναι ορατό στο σχήμα 4.1.



Σχήμα 4.1: Αντιστοίχηση λέξεων κλειδιών σε έννοιες και βάρη.

Όπως μπορούμε να δούμε, για κάθε κείμενο που εμφανίζεται στη συλλογή μας, μπορούμε να το κατατάξουμε σε μία κατηγορία βάση της έννοιας την οποία εμπεριέχει ή σε κάποια κατηγορία ανάλυσης ανάλογα με τη διαδικασία που ακολουθήσαμε για να επεξεργαστούμε την πληροφορία του κειμένου. Αυτό θα γίνει πιο σαφές μέσα από ένα παράδειγμα. Μπορούμε σε ένα κείμενο να κάνουμε μια εκτενέστατη ανάλυση και να εντοπίσουμε πως αναφέρεται στην κατηγορία ποδόσφαιρο από τη γενικότερη κατηγορία αθλητικά. Το βάρος που θα έχει για το κείμενο η έννοια ποδόσφαιρο θα είναι μεγάλη σε αυτή την ανάλυση, όμως θα περιοριστεί με αυτό τον τρόπο το κοινό στο οποίο απευθύνεται το κείμενο καθώς έχουμε μικρό βαθμό αφαίρεσης. Από την άλλη μεριά μπορούμε να κάνουμε μια πιο γενική ανάλυση του κειμένου, εντοπίζοντας απλά την κατηγορία στην οποία ανήκουν ενδεικτικές προτάσεις μέσα από το κείμενο. Με αυτό τον τρόπο οι λέξεις κλειδιά μέσα στα κείμενα ταξινομούνται βάσει του αφαιρετικού βάρους, δηλαδή βάση της ικανότητάς τους να περιγράψουν το κείμενο σαν λέξεις που επελέγησαν από μέρος του κειμένου και όχι από το σύνολό του. Τα κομμάτια που επελέγησαν από το κείμενο μπορούν να αποτελέσουν και μια σύντομη περιγραφή του κειμένου, η αλλιώς μια περίληψή του. Ως εκ' τούτου, φαίνεται να δημιουργούνται δύο είδη γενικών κατηγοριών. Πρόκειται για τις κατηγορίες που δημιουργούνται για τους χρήστες που επιθυμούν να έχουν πρόσβαση σε πολύ εξειδικευμένη πληροφορία και σε αυτούς που επιθυμούν να έχουν πρόσβαση σε πληροφορία γενικά.

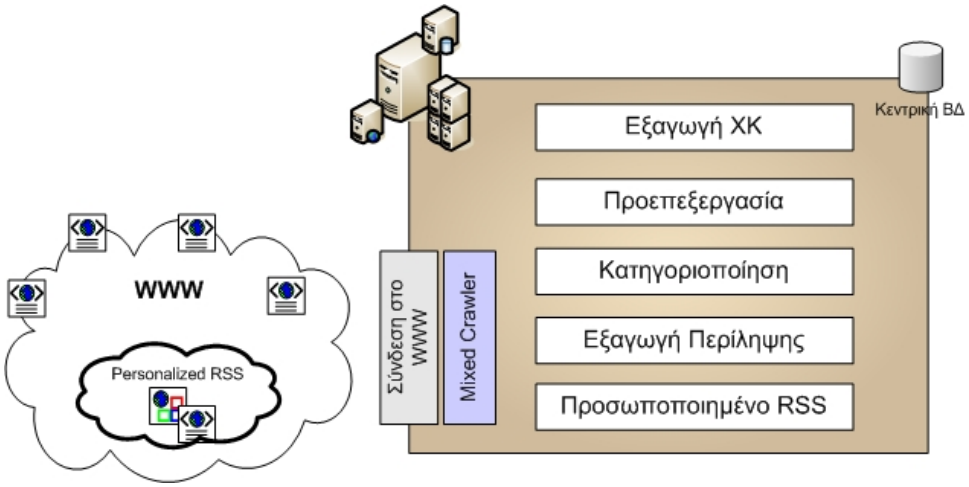
Όσον αφορά την έννοια της ανάλυσης είναι ένα κομμάτι το οποίο θα γίνεται αλγοριθμικά βασισμένο στη μέθοδο Support Vector Machines . Είναι αναπόφευκτη η χρήση της μηχανής στο συγκεκριμένο κομμάτι. Ωστόσο το θέμα που αφορά το αφαιρετικό βάρος δε θα μπορούσε να ανήκει αποκλειστικά σε μία μηχανή αλλά περισσότερο σε έναν άνθρωπο, σε ένα χρήστη του συστήματος. Οι χρήστες θα είναι αυτοί που θα έχουν τον κύριο λόγο στη δημιουργία των αφαιρετικών βαρών ανάλογο με τον τρόπο με τον οποίο αντιμετωπίζουν τα αποτελέσματα. Στόχος είναι να ενταχθεί ο χρήστης του συστήματος στη διαδικασία με την οποίο λαμβάνει ο ίδιος αποτελέσματα, είτε αυτά αφορούν τη διαδικασία της περίληψης, είτε τη διαδικασία της κατηγοριοποίησης. Απώτερος στόχος είναι να γίνει ξεχωριστή περίληψη και κατηγοριοποίηση της πληροφορίας για κάθε χρήστη, προκειμένου ο καθένας ο οποίος χρησιμοποιεί το σύστημα να είναι σε θέση να έρθει πιο κοντά στα αποτελέσματα που επιθυμεί να βρει.

4.2 Γενική αρχιτεκτονική του συστήματος

Η αρχιτεκτονική του συστήματος βασίζεται σε κατανεμημένη αρχιτεκτονική και αυτόνομα υποσυστήματα, αλλά η διαδικασία που ακολουθείται για να παραχθεί το επιθυμητό αποτέλεσμα είναι στην ουσία ακολουθιακή. Αυτό σημαίνει ότι η ροή πληροφορίας είναι αντιπροσωπευτική των υποσυστημάτων από τα οποία αποτελείται το σύστημα. Ένα άλλο σημαντικό αρχιτεκτονικό ζήτημα είναι η τμηματικότητα του μηχανισμού.

Εστιάζουμε στο τμήμα της περίληψης κειμένου παρότι θα παρουσιαστεί αναλυτικά το τμήμα κατηγοριοποίησης καθώς και το τμήμα προσωποποίησης του μηχανισμού, με στόχο να παρουσιαστεί η αλληλοσύνδεση των τμημάτων αυτών του συνολικού συστήματος. Όπως έχει ήδη αναφερθεί, η διαδικασία περίληψης δέχεται πληροφορία από την διαδικασία προ-επεξεργασίας και ανταλλάσσει γνώση με τους μηχανισμούς κατηγοριοποίησης και προσωποποίησης, με στόχο την δημιουργία της περίληψης κειμένου σύμφωνα με τις ανάγκες του κάθε χρήστη.

Ο μηχανισμός αποτελείται από μια σειρά από συστήματα για την παραγωγή του επιθυμητού αποτελέσματος. Η συνεργασία μεταξύ των κατανεμημένων υποσυστημάτων βασίζεται σε ανοιχτά πρότυπα για είσοδο και έξοδο τα οποία υποστηρίζονται από κάθε τμήμα του συστήματος αλλά και από την επικοινωνία με την κεντρικοποιημένη βάση δεδομένων. Το σχήμα 4.2 παρουσιάζει την αρχιτεκτονική του συνολικού μηχανισμού.



Σχήμα 4.2: Βασική Αρχιτεκτονική του Συστήματος.

Ο μηχανισμός, όπως παρουσιάζεται στο σχήμα 4.2 ακολουθεί τις παρακάτω διαδικασίες:

I. Συλλογή ιστοσελίδων που περιέχουν ειδήσεις από τον Παγκόσμιο Ιστό και εξαγωγή του χρήσιμου κείμενου από αυτές

II. Ανάλυση του εξαγόμενου κείμενου

III. Εφαρμογή αλγορίθμων κατηγοριοποίησης και περίληψης στο κείμενο

IV. Εφαρμογή αλγορίθμων προσωποποίησης για τον εκάστοτε χρήστη στο κείμενο

V. Παρουσίαση των αποτελεσμάτων ως έξοδο στον χρήστη συσκευής μικρού μεγέθους ή σε κάποιο άλλο σύστημα που ακολουθεί

Για να συλλεχθούν οι ιστοσελίδες από τον παγκόσμιο ιστό, ένας απλός εστιασμένος Crawler[1] χρησιμοποιείται. Οι διευθύνσεις οι οποίες χρησιμοποιούνται ως είσοδος για τον crawler εξάγονται από ροές νέων (RSS Feeds). Οι ροές νέων 'δείχνουν' απευθείας σε σελίδες όπου υπάρχουν τα άρθρα με νέα. Ο crawler

[1]Focused Crawler: Ένας εστιασμένος μηχανισμός αυτόματης αναζήτησης και καταγραφής περιεχομένου του διαδικτύου που σε αντίθεση με έναν απλό Crawler δεν διαπερνά ότι βρεθεί στο δρόμο του, παρά σελίδες που είναι σχετικές με ένα προκαθορισμένο σετ θεμάτων

αποθηκεύει τις ιστοσελίδες σε μορφή html χωρίς άλλα στοιχεία της ιστοσελίδας (εικόνες, css, javascript, κ.λπ. παραβλέπονται). Αποθηκεύοντας μόνο την σελίδα σε μορφή html, η βάση δεδομένων γεμίζει με σελίδες που είναι έτοιμες για είσοδο στο πρώτο επίπεδο ανάλυσης.

Η 'έξοδος' από έναν crawler είναι συχνά καθαρός κώδικας HTML χωρίς καμία επεξεργασία. Φυσικά η χρήση του κώδικα στη διαδικασία κατηγοριοποίησης ή περίληψης είναι κάτι το απαγορευτικό. Προκειμένου λοιπόν να μπορέσουμε να προχωρήσουμε στο επόμενο βήμα θα πρέπει να έχουμε στα χέρια μας καθαρό κείμενο. Ένας μηχανισμός ανάλυσης και εξαγωγής του χρήσιμου μόνο κειμένου από σελίδες του διαδικτύου κατασκευάστηκε, προκειμένου να μπορέσουμε να παρέχουμε στις διαδικασίες του συστήματος 'καθαρό' κείμενο (χωρίς στοιχεία κώδικα). Το εξαγόμενο 'καθαρό' κείμενο αποθηκεύεται επίσης στη βάση απ' όπου χρησιμοποιείται ασύγχρονα από τα επόμενα βήματα.

Αρχικό και βασικό χαρακτηριστικό του συστήματος είναι το σύνολο των κειμένων εκπαίδευσης. Προκειμένου να είναι εφικτή η δυνατότητα αυτόματης κατηγοριοποίησης θα πρέπει να αρχικοποιηθούν κάποιες βασικές κατηγορίες με κείμενα αντιπροσωπευτικά αυτών. Έτσι το κομμάτι εκείνο το οποίο θα είναι υπεύθυνο για την κατηγοριοποίηση των κειμένων, θα πρέπει αρχικά να αναλάβει να δημιουργήσει τις διαφορετικές κατηγορίες. Εν συνεχεία θα είναι σε θέση να παραλάβει μη κατηγοριοποιημένα κείμενα και να προσπαθήσει να τα εντάξει σε κάποια από τις ήδη υπάρχουσες κατηγορίες.

Η κατηγοριοποίηση της πληροφορίας περνά από συγκεκριμένα στάδια τα οποία αποτελούν και διαφορετικά υποσυστήματα που λειτουργούν σειριακά για κάθε ξεχωριστό κείμενο αλλά συνολικά παράλληλα. Έτσι υπάρχει ξεχωριστός μηχανισμός που πραγματοποιεί την προεπεξεργασία και ξεχωριστός μηχανισμός που αναλαμβάνει να 'τρέξει' τον αλγόριθμο κατηγοριοποίησης για κάθε επεξεργασμένο κείμενο.

Κατά τη διάρκεια του πρώτου επιπέδου ανάλυσης, το σύστημα μας απομονώνει το 'χρήσιμο κείμενο' από την html σελίδα. Ως χρήσιμο κείμενο κρατάμε τον τίτλο και το κυρίως σώμα του άρθρου (article body). Το δεύτερο επίπεδο ανάλυσης δέχεται ως είσοδο αρχεία σε μορφή XML τα οποία περιλαμβάνουν τον τίτλο και το σώμα των άρθρων. Ο κύριος σκοπός του είναι να εφαρμόσει αλγορίθμους προ-επεξεργασίας πάνω στο κείμενο και να παράγει ως έξοδο λέξεις-κλειδιά, την θέση τους στο κείμενο καθώς και την συχνότητα εμφάνισης τους μέσα σε αυτό. Αυτά τα αποτελέσματα είναι απαραίτητα για να προχωρήσουμε στο τρίτο επίπεδο ανάλυσης.

Η καρδία του μηχανισμού μας βρίσκεται στο τρίτο επίπεδο ανάλυσης, όπου τα υποσυστήματα της περίληψης και κατηγοριοποίησης εντοπίζονται. Ο κύριος στόχος τους είναι να χαρακτηρίζουν ένα άρθρο με μία ετικέτα (κατηγορία) και να παράγουν μια περίληψή του. Τα αποτελέσματα μπορούν στη συνέχεια είτε να παρουσιαστούν στους τελικούς χρήστες μέσω ενός προσωποποιημένου portal, είτε να οδηγηθούν προς χρήση σε άλλα υποσυστήματα που ακολουθούν. Η έξοδος αυτή ακολουθεί επίσης τα ανοιχτά πρότυπα και δίνεται σε μορφή XML και επομένως εύκολα αναγνωρίσιμη από οποιοδήποτε υποσύστημα μπορεί να ακολουθεί.

Οι παραπάνω ξεχωριστοί μηχανισμοί όπως προαναφέρθηκε πρέπει να λειτουργήσουν σειριακά πάνω σε κάθε ξεχωριστό κείμενο προκειμένου να είναι επιτυχημένη τόσο η κατηγοριοποίηση όσο και η δημιουργία του δυναμικού προφίλ. Ωστόσο κάθε μηχανισμός εσωτερικά είναι δημιουργημένος ώστε να μπορεί να δουλεύει σαν ένα παράλληλο σύστημα αφού καθένας είναι ανεξάρτητος από τους υπολοίπους. Μπορούμε π. χ. να εκτελούμε το crawling και το κατέβασμα των σελίδων σε διαφορετική χρονική στιγμή από το βήμα της προεπεξεργασίας κειμένου για εξαγωγή κωδικολέξεων γλιτώνοντας έτσι πολύτιμο χρόνο που αφορά στο κατέβασμα της σελίδας. Τέλος, όλοι οι μηχανισμοί που έχουν αναπτυχθεί, χρησιμοποιούν τοπικά μία μικρή μνήμη του συστήματος (κατά την εκτέλεσή τους) προκειμένου να αποθηκεύουν (τοπικά) συγκεκριμένα αποτελέσματα από τις διαδικασίες τους, η μνήμη αυτή περιλαμβάνει είτε τη φυσική μνήμη συστήματος που καταλαμβάνουν τα προγράμματα κατά την εκτέλεσή τους, είτε κάποια προσωρινά αρχεία. Ωστόσο, όλα τα αποτελέσματα συγκεντρώνονται σε μία κεντρικοποιημένη βάση δεδομένων, προκειμένου να εξασφαλιστεί η ακεραιότητα τους αλλά και η διαθεσιμότητα τους όσο το δυνατόν νωρίτερα στους υπόλοιπους μηχανισμούς. Η κεντρική βάση δεδομένων μπορεί να προκαλεί αρκετή καθυστέρηση σε συγκεκριμένα κομμάτια του συστήματος, ωστόσο μιλούμε για ένα σύστημα το οποίο απαιτεί απόλυτη ακρίβεια στα δεδομένα και αποφυγή διπλοεγγραφών η σφαλμάτων.

Λίγο πριν την παρουσίαση των αποτελεσμάτων στο χρήστη, υπάρχει ο μηχανισμός ο οποίος αναλαμβάνει να διαχειριστεί το προφίλ του κάθε χρήστη ούτως ώστε να παράγει το προσωποποιημένο περιεχόμενο που θα του αποσταλεί. Πρόκειται για ένα μηχανισμό ο οποίος λαμβάνει υπόψη του τις προτιμήσεις του χρήστη όσον αφορά τις κατηγορίες νέων ή κάποια ξεχωριστής σημασίας λέξεων-κλειδιών που τον ενδιαφέρουν, αλλά και τις δυνατότητες απεικόνισης που διαθέτει η συσκευή του, π. χ. πρόκειται για pda, κινητό τηλέφωνο, φορητό υπολογιστή ή για κάποια άλλη συσκευή. Οι πληροφορίες αυτές οργανώνονται και αποθηκεύονται

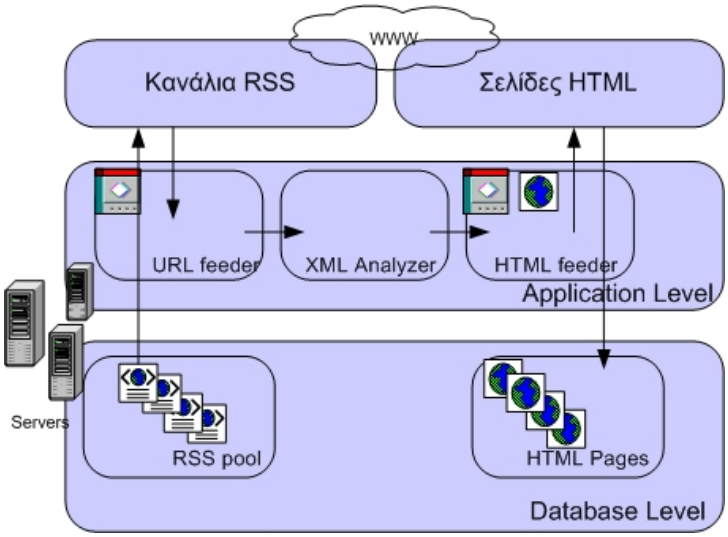
επίσης στην κεντρικοποιημένη βάση δεδομένων, ώστε να κατασκευαστεί το δυναμικό προφίλ του χρήστη.

4.3 Υποσυστήματα

Στη συνεχεία παρουσιάζονται τα υποσυστήματα του μηχανισμού προκειμένου να γίνει κατανοητή η λειτουργία του μηχανισμού σε κάθε διαφορετικό επίπεδο υλοποίησης.

4.3.1 Συλλογή πληροφορίας

Για τη συλλογή πληροφορίας για το σύστημά μας και πιο συγκεκριμένα για την συνεχή και αδιάκοπη συλλογή άρθρων από το Διαδίκτυο εκμεταλλευόμαστε την τάση που επικρατεί σε όλους τους δικτυακούς τόπους να προσφέρουν κανάλια άμεσης επικοινωνίας με τους χρήστες, και δε μιλούμε για κάτι διαφορετικό από τα RSS.

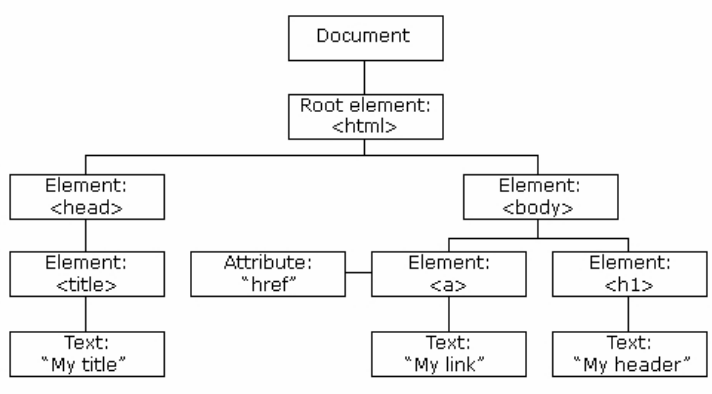


Σχήμα 4.3: Μηχανισμός Συλλογής Πληροφορίας.

Η αρχιτεκτονική του μηχανισμού είναι απλή και εκμεταλλεύεται πλήρως το γεγονός ότι οι μεγαλύτερες δικτυακές πύλες ενημέρωσης προσφέρουν στους χρήστες RSS feeds. Όπως φαίνεται από το σχήμα 4.3, ένας απλοϊκός mixed selective crawler χρησιμοποιείται προκειμένου να λαμβάνει το σύστημά μας HTML σελίδες. Πρόκειται για έναν mixed crawler διότι συνδυάζει τη χρήση wrapper και crawler. Ο wrapper είναι ένας μηχανισμός αναγνώρισης προτύπων που συνήθως ακολουθείται από επεξεργασία αυτών. Στην περίπτωσή μας ο wrapper στο μηχανισμό συλλογής πληροφορίας εντοπίζει μέσα στα XML αρχεία εκείνα τα σημεία τα οποία περιέχουν πληροφορίες για τα άρθρα που θέλουμε να εξάγουμε. Μέσα από αυτά τα αρχεία προκύπτουν τα URL seeds τα οποία επανατροφοδοτούν το ίδιο μηχανισμό για να προχωρήσει στο 'κατέβασμα' των σελίδων HTML, που περιέχουν άρθρα, από τη φυσική τους θέση χωρίς να χρειαστεί καμία απολύτως αναζήτηση. Ο wrapper συνεπώς χρησιμοποιείται για να μπορέσουμε να εξάγουμε τον τίτλο του άρθρου και τη διεύθυνση στην οποία βρίσκεται με τη βοήθεια των RSS feeds και εν συνεχεία το πρόγραμμα αλλάζει μορφή και μετατρέπεται σε crawler ο οποίος 'επισκέπτεται' τα URLs που έχει εξάγει ο wrapper και από αυτά λαμβάνει τον HTML κώδικα. Η βάση δεδομένων δε χρειάζεται τις ενδιάμεσες πληροφορίες και έτσι οι πληροφορίες που έχει είναι η λίστα με τα RSS. Τις πληροφορίες που αποθηκεύονται για κάθε άρθρο θα τις δούμε στη συνέχεια του κεφαλαίου.

4.3.2 Φιλτράρισμα Χρήσιμου κειμένου

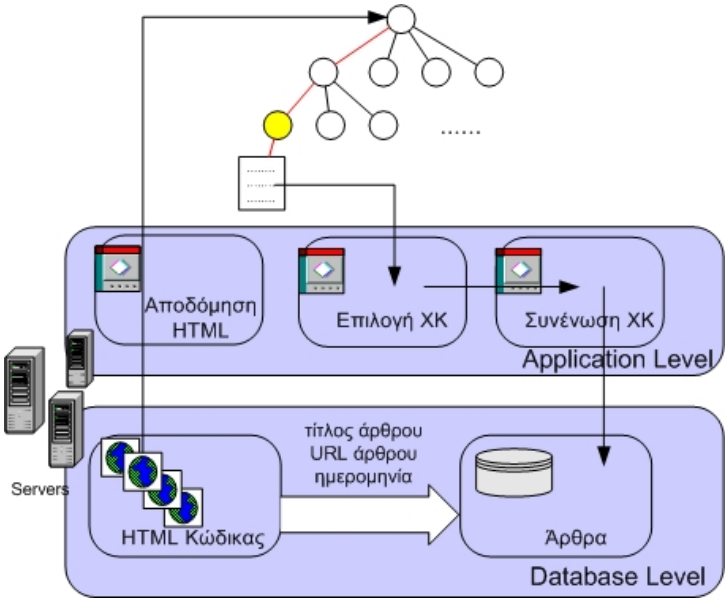
Για την εξαγωγή του χρήσιμου κειμένου χρησιμοποιείται η ιδιότητα της HTML να μπορεί να αναπαρασταθεί σε δενδρική μορφή σύμφωνα με το DOM (Document Object Model) μοντέλο, όπως φαίνεται και στο σχήμα 4.4.



Σχήμα 4.4: *HTML Document Object Model* (*DOM*).

Το σχήμα 4.4 είναι η DOM αναπαράσταση του παρακάτω HTML κώδικα.

```
<html>
  <head>
    <title>My Title</title>
  </head>
  <body>
    <a href="#">My Link</a>
    <h1>My Header</h1>
  </body>
</html>
```



Σχήμα 4.5: Εξαγωγή Χρήσιμου Κειμένου.

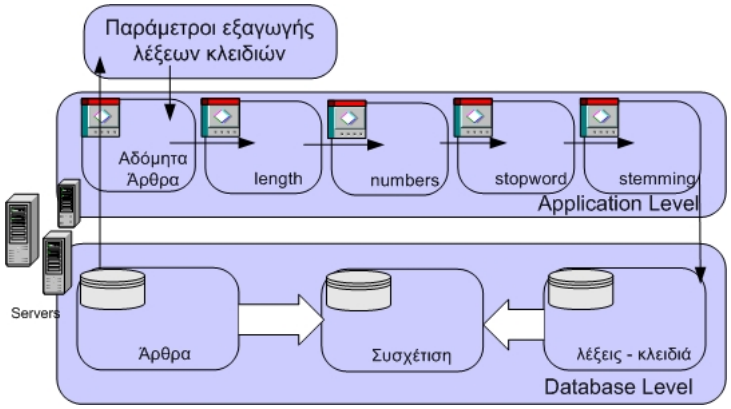
Βασιζόμενοι λοιπόν στο γεγονός ότι κάθε HTML κώδικας μπορεί να αποδομηθεί στα βασικά του στοιχεία σε δενδρική μορφή, χρησιμοποιούμε ένα μηχανισμό όπως αυτός που φαίνεται στο σχήμα 4.5, προκειμένου να εξάγουμε το χρήσιμο κείμενο από τις HTML σελίδες.

Και πάλι σε αυτή την περίπτωση εργαζόμαστε σε δύο επίπεδα αυτό της εφαρμογής και αυτό της Βάσης Δεδομένων. Από τη ΒΔ λαμβάνουμε τον HTML κώδικα καθώς και πληροφορίες για το άρθρο που έχουν συλλεχθεί από το προηγούμενο στάδιο και προχωρούμε σε αποδόμηση της HTML σελίδας προκειμένου να

εντοπίσουμε τα φύλλα του δένδρου που ενδεχόμενα περιέχουν χρήσιμες πληροφορίες για το μηχανισμό.

4.3.3 Προεπεξεργασία κειμένου

Το υποσύστημα προεπεξεργασίας κειμένου είναι ένα σημαντικό τμήμα του συνολικού μηχανισμού το οποίο αναλαμβάνει το καθάρισμα του σώματος του κειμένου και την εξαγωγή κωδικολέξεων (keywords). Η διαδικασία για την προεπεξεργασία κειμένου και την εξαγωγή των λέξεων κλειδιών φαίνεται στο Σχήμα 4.6. Η είσοδος στο υποσύστημα αυτό είναι μορφής XML που περιέχει τα απαραίτητα μόνο στοιχεία: τίτλος και σώμα κειμένου.



Σχήμα 4.6: Προεπεξεργασία κειμένου και εξαγωγή κωδικολέξεων.

Εκτός από το αρχείο (ή τη δομή) XML, σαν είσοδος στον μηχανισμό δίνεται ένας αριθμός από παραμέτρους:

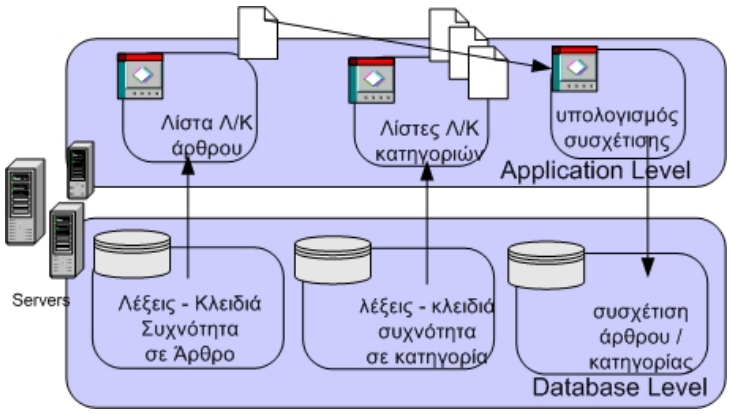
- το ελάχιστο μήκος λέξης (οι λέξεις που είναι μικρότερες από αυτό το μήκος θα αφαιρεθούν)
- καθορισμός εάν τα αριθμητικά δεδομένα θα κρατηθούν ή θα αφαιρεθούν
- καθορισμός μιας λίστας από λέξεις τετριμμένες και συνηθισμένες οι οποίες δεν εκφράζουν κάποιο συγκεκριμένο νόημα και μπορούν να θεωρηθούν ως 'σκουπίδια' (stopwords)
- καθορισμός του αλγορίθμου stemming που θα χρησιμοποιηθεί

Η έξοδος του μηχανισμού προεπεξεργασίας κειμένου μπορεί είτε να αποθηκεύεται στη βάση δεδομένων του συστήματος, είτε να δρομολογείται σε άλλα υποσυστήματα που ακολουθούν. Στην δεύτερη περίπτωση, η έξοδος έχει τη μορφή XML αρχείου έτσι ώστε να είναι εύκολη η διασύνδεση με άλλα υποσυστήματα. Η έξοδος περιλαμβάνει:

- τις κωδικολέξεις που προέκυψαν από την διαδικασία του keyword extraction
- τις θέσεις των keywords στο αρχικό κείμενο, σε ποιες προτάσεις δηλαδή εμφανίζονται
- το πλήθος με το οποίο εμφανίζονται τα keywords κάτι που εκφράζεται είτε ως απόλυτη συχνότητα εμφάνισης (π. χ. ένα keyword εμφανίζεται 5 φορές στο κείμενο), είτε ως σχετική συχνότητα εμφάνισης (π. χ. ένα keyword εμφανίζεται 5 φορές σε ένα κείμενο 50 λέξεων, άρα με σχετική συχνότητα 0,1).

4.3.4 Κατηγοριοποίηση Κειμένου

Το υποσύστημα της κατηγοριοποίησης κειμένου αποτελεί ένα κεντρικό συστατικό του μηχανισμού που αναπτύχθηκε και σε συνδυασμό με εκείνο της εξαγωγής περίληψης, βρίσκονται στο δεύτερο επίπεδο ανάλυσης του συστήματος αποτελώντας τον πυρήνα του μηχανισμού. Το υποσύστημα περιγράφεται από το Σχήμα 4.7.



Σχήμα 4.7: Μηχανισμός κατηγοριοποίησης κειμένου.

Η είσοδος του υποσύστηματος κατηγοριοποίησης κειμένου είναι XML αρχεία τα οποία περιέχουν την έξοδο του υποσυστήματος εξαγωγής κωδικολέξεων και πιο συγκεκριμένα: τα keywords του κειμένου και τις συχνότητες εμφάνισής τους στο κείμενο. Ο βασικός στόχος του υποσυστήματος αυτού είναι η εφαρμογή αλγορίθμων κατηγοριοποίησης στο κείμενο και επομένως η αντιστοίχηση του κειμένου με κάποια από τις ήδη υπάρχουσες κατηγορίες. Βασικό ρόλο σε αυτή τη διαδικασία παίζει η ύπαρξη μιας σωστής, πλήρης και αποτελεσματικής βάσης γνώσης πάνω στην οποία θα στηρίζεται η κατηγοριοποίηση. Πιο αναλυτικά, χρειαζόμαστε κάποιες βασικές κατηγορίες άρθρων, στις οποίες θα εμπίπτουν τα περισσότερα των νέων άρθρων που έρχονται στο σύστημα, καθώς και ένα πλήθος αντιπροσωπευτικών της κάθε κατηγορίας κειμένων, τα οποία έχουν περάσει από το μηχανισμό εξαγωγής keywords και στην ουσία 'ταΐζουν' το σύστημα με την αναγκαία γνώση, ώστε να μπορεί με χρήση απλών μετρικών να κατηγοριοποιεί νεοαφιχθέντα άρθρα.

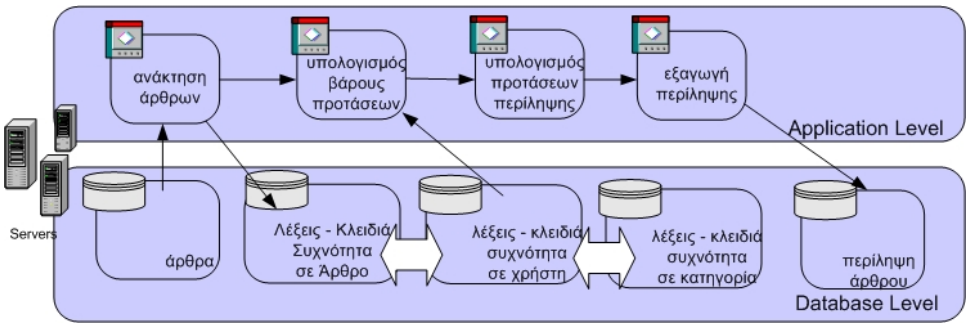
Το υποσύστημα κατηγοριοποίησης βασίζεται στην μετρική ομοιότητας συνημιτόνου, σε εσωτερικά γινόμενα καθώς και σε υπολογισμούς ζυγίσματος όρων. Η χρήση αυτών των μετρικών γίνεται ύστερα από την αρχικοποίηση του training set της βάσης γνώσης και μέσω μιας διαδικασίας η οποία on the fly ελέγχει τη συσχέτιση του κάθε keyword του προς κατηγοριοποίηση κειμένου με τις υπάρχουσες κατηγορίες. Οι συσχετίσεις που θα βρεθούν αθροίζονται και κανονικοποιούνται με αποτέλεσμα να προκύπτει για κάθε κείμενο ένα ποσοστό ομοιότητας (relativity) με κάθε μια από τις υπάρχουσες κατηγορίες. Εάν το training set είναι αποτελεσματικό και αξιόπιστο, τα άρθρα που περιέχουν πολλά keywords σχετικά με κάποια από τις κατηγορίες, θα πρέπει να έχουν κατηγοριοποιηθεί με συσχέτιση μεγαλύτερη ως προς αυτή. Στην πράξη βέβαια, ακόμη και αν το κείμενο είναι εντελώς αντιπροσωπευτικό κάποιας κατηγορίας, δεν αποκτά συσχέτιση 100% με μία και μόνο κατηγορία, αφού είναι φυσικό να περιέχει ορισμένα keywords τα οποία συσχετίζονται και με τις υπόλοιπες κατηγορίες (το άθροισμα των συσχετίσεων ενός κειμένου με όλες τις κατηγορίες, προφανώς σε κάθε περίπτωση είναι 1).

Η έξοδος του υποσυστήματος κατηγοριοποίησης, οι συσχετίσεις δηλαδή του κειμένου με κάθε κατηγορία, αποθηκεύονται στη βάση δεδομένων του συστήματος.

4.3.5 Εξαγωγή Περίληψης Κειμένου

Το υποσύστημα εξαγωγής περίληψης (Σχήμα 4.8) κειμένου του μηχανισμού αποτελεί ένα ανεξάρτητο υποσύστημα το οποίο δέχεται ως είσοδο τα αποτελέσματα του keyword extraction (αποθηκευμένα στη βάση ή σε μορφή XML) που περιέχουν: τα keywords που κρατήθηκαν, τη συχνότητα εμφάνισής τους στο κείμενο, τις θέσεις τους (σε ποιες προτάσεις εμφανίζονται, π. χ. 1η, 3η, κ.ο.κ.) και το πόσες προτάσεις πρέπει να κρατηθούν για την τελική περίληψη. Τα στοιχεία αυτά, μαζί με την πληροφορία για τον τίτλο του κειμένου, είναι αρκετά ώστε να μπορεί το υποσύστημα αυτό να επιχειρεί μια βαθμολόγηση των προτάσεων του κειμένου. Θα πρέπει να πούμε σε αυτό το σημείο ότι, ο μηχανισμός αυτόματης εξαγωγής περίληψης δεν χρειάζεται απαραίτητα αυτό καθ' αυτό το κείμενο αν και για να παραχθεί η τελική περίληψη ενός κειμένου αυτό είναι αναγκαίο. Με το προηγούμενο εννοούμε ότι, το υποσύστημα αυτό μπορεί να παράγει μια τελική κατάταξη των προτάσεων του κειμένου απλά και μόνο με τις εισόδους που περιγράφηκαν νωρίτερα και ενώ το αρχικό κείμενο βρίσκεται αποθηκευμένο μία φορά μόνο στην βάση δεδομένων. Το τελευταίο δεδομένο εισόδου

του υποσυστήματος περιγράφει πόσες προτάσεις επιθυμούμε να έχουμε ως έξοδο για περίληψη του αρχικού κειμένου. Το πλήθος των προτάσεων μπορεί να καθοριστεί είτε ως ποσοστό % των προτάσεων του αρχικού κειμένου είτε ως συνολικό πλήθος χαρακτήρων. Για παράδειγμα, αν το αρχικό κείμενο είχε 20 προτάσεις και κρατάμε ένα ποσοστό 30% επί των προτάσεων, στην περίληψη θα κρατηθούν οι 6 σημαντικότερες προτάσεις του κειμένου, αντίθετα, εάν επιθυμούμε η περίληψη του κειμένου να περιέχει περίπου ένα συγκεκριμένο πλήθος χαρακτήρων, θα επιλεχθούν τόσες προτάσεις από τις σημαντικότερες ώστε και να καλύπτεται το πλήθος χαρακτήρων που τέθηκε και να μην ξεπερνιέται κατά πολύ αυτό. Στην ουσία επιλέγεται η βέλτιστη επιλογή μήκους χαρακτήρων στο όριο να επιλεχθεί μια παραπάνω πρόταση ή μια λιγότερη.



Σχήμα 4.8: Μηχανισμός περίληψης κειμένου.

Η έξοδος επομένως του υποσυστήματος αυτόματης εξαγωγής περίληψης κειμένου είναι μια φθίνουσα σειρά προτάσεων με βάση το σκορ που αξιολογεί ο μηχανισμός πως πρέπει να έχουν όσον αφορά την σημαντικότητά τους για να αναπαραστήσουν το κείμενο. Η βαθμολόγηση των προτάσεων του κειμένου γίνεται βάσει των keywords όπου αυτές περιέχουν και αφορά στις παρακάτω σημαντικές παραμέτρους:

- υπάρχει το keyword και στον τίτλο του κείμενου;
- υπάρχει πληροφορία για την κατηγορία που ανήκει το κείμενο;
- υπάρχει πληροφορία για τις προτιμήσεις του χρήστη σε κατηγορία ή keywords;

Το ζύγισμα των παραπάνω παραμέτρων είναι κεφαλαιώδους σημασίας για τον μηχανισμό αυτόματης εξαγωγής περίληψης καθώς η εύρεση των βέλτιστων παραγόντων που θα χρησιμοποιηθούν θα κρίνει και το σκορ που θα λάβουν οι προτάσεις, επομένως και την περίληψη του κειμένου.

Ένα άλλο σημαντικό θέμα είναι η σειρά εμφάνισης των προτάσεων στην τελική περίληψη που προκύπτει. Είναι πιθανό, προτάσεις που βρίσκονται όχι στην αρχή του κειμένου να είναι πιο αντιπροσωπευτικές του νοήματος του κειμένου και επομένως να λαμβάνουν υψηλότερο σκορ από το μηχανισμό σε σχέση με άλλες οι οποίες βρίσκονται νωρίτερα στο κείμενο. Η παρουσίαση όμως τυχαίων προτάσεων στον τελικό χρήστη, κάθε άλλο παρά κατανοητή περίληψη είναι. Είναι σωστότερο επομένως, αφού έχει επιλεχθεί το πλήθος των προτάσεων που θα απαρτίζουν μια περίληψη, να γίνει μια ταξινόμησή τους σε σχέση με τη σειρά εμφάνισής τους στο κείμενο, διατηρώντας έτσι τη νοηματική συνοχή του κειμένου πριν παρουσιαστούν στον τελικό χρήστη.

4.3.6 Παρουσίαση πληροφορίας και προσωποποίηση στο χρήστη

Η προσωποποίηση του περιεχομένου (περίληψη) στον χρήστη περιλαμβάνει δύο επίπεδα:

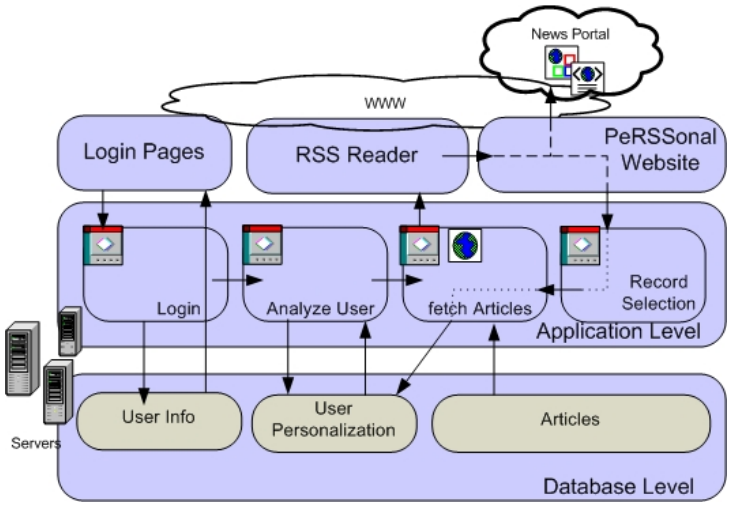
- προσωποποίηση λόγω προτιμήσεων του χρήστη σε συγκεκριμένες κατηγορίες άρθρων
- προσωποποίηση λόγω περιορισμών της συσκευής προβολής του τελικού χρήστη

Απώτερος σκοπός του υποσυστήματος προσωποποίησης είναι ο χρήστης να μην αντιλαμβάνεται όλες τις διεργασίες που λαμβάνουν χώρα και να απολαμβάνει ποιοτικά και γρήγορα αποτελέσματα βάση των προσωπικών του επιλογών. Για την προσωποποίηση στο χρήστη μπορούν να χρησιμοποιηθούν δύο μέθοδοι:

1. Ο χρήστης να δώσει κάποια πληροφορία στο σύστημα και το σύστημα να ξεκινήσει παρουσιάζοντας εξ αρχής προσωποποιημένα αποτελέσματα και να συγκλίνει γρήγορα στις ανάγκες του χρήστη.

2. Ο χρήστης να μη δώσει καθόλου πληροφορία στο σύστημα και το σύστημα να ξεκινήσει παρουσιάζοντας γενικές πληροφορίες και να αργήσει να συγκλίνει στις προσωπικές επιλογές του χρήστη.

Σε κάθε περίπτωση το επιθυμητό επιτυγχάνεται και πρόκειται για τη σύγκλιση των πληροφοριών που παρουσιάζονται στις ανάγκες του χρήστη. Το σχήμα 4.9 απεικονίζει την αρχιτεκτονική του συγκεκριμένου μηχανισμού. Ο χρήστης, εφόσον έχει εγγραφεί στο σύστημα, ζητώντας το προσωπικό του RSS ουσιαστικά 'συνδέεται' με το σύστημα. Αυτό δέχεται την αίτησή του και εξάγει τις προσωπικές του προτιμήσεις από τη ΒΔ. Στη συνέχεια, και βάσει των προτιμήσεων του χρήστη, ανακτά τα άρθρα που τον ενδιαφέρουν και επιτελεί την εξαγωγή προσωποποιημένης περίληψης πάνω σε αυτά· πλέον το RSS είναι έτοιμο για αποστολή στον χρήστη. Στη συνέχεια, και λόγω της ανάγκης για συνεχή ανανέωση των επιλογών και προτιμήσεων του χρήστη, το σύστημα καταγράφει τις επισκέψεις των χρηστών στα άρθρα που του δόθηκαν μέσω του RSS. Αυτό γίνεται ως εξής: στο RSS που δίνεται ως απάντηση στον χρήστη, τα links που περιέχουν την πηγή του περιληπτημένου άρθρου ανακατευθύνονται μέσω του συστήματος μέσω απλών σελίδων PHP. Η ανακατεύθυνση αυτή (redirect) των χρηστών μπορεί να μας δώσει στοιχεία για το ποια άρθρα από το RSS feed αποφάσισε να επισκεφθεί ο χρήστης και πότε, δίνοντάς μας έτσι την δυνατότητα διαρκούς ανανέωσης του δυναμικού προφίλ του. Αποτέλεσμα αυτού είναι το σύστημα, ακόμη και αν έχει ξεκινήσει από ένα προφίλ εντελώς άσχετο για κάποιον χρήστη (λόγω π. χ. εσκεμμένα λανθασμένης βαθμολόγησης κατηγοριών), να μπορεί να συγκλίνει πολύ γρήγορα στο πραγματικό προφίλ που εκφράζει τον χρήστη. Τα ζητήματα της προσωποποίησης της περίληψης στο χρήστη, αλλά και της διαρκούς ανανέωσης του προφίλ του περιγράφονται σχηματικά από το υποσύστημα peRSSonal του Σχήματος 4.9.



Σχήμα 4.9: Αρχιτεκτονική της προσωποποίησης των περιλήψεων στον χρήστη.

Η προσωποποιημένη πληροφορία που προέκυψε από το προηγούμενο βήμα, αποστέλλεται στο χρήστη της συσκευής μικρού μεγέθους μέσω του προτύπου RSS και προβάλλεται με τον καλύτερο δυνατό τρόπο στον RSS Reader της συσκευής του. Ο τρόπος απεικόνισης έχει εξασφαλιστεί από το προηγούμενο στάδιο όταν η προσωποποίηση έγινε και με βάση το πλήθος των χαρακτήρων που μπορεί να απεικονίσει η συσκευή του χρήστη. Με αυτό τον τρόπο, ο χρήστης π. χ. ενός κινητού τηλεφώνου δεν είναι αναγκασμένος να κάνει scrolling διαρκώς για να διαβάσει μια περίληψη αφού αυτή έχει το κατάλληλο μέγεθος για το ελάχιστο δυνατό scrolling, δεδομένων των παραμέτρων της τελικής συσκευής που λήφθηκαν υπ' όψιν κατά τη διαδικασία της προσωποποίησης της περίληψης στον χρήστη.

Φυσικά ο μηχανισμός δύναται να παρουσιάζει προσωποποιημένο περιεχόμενο περιλήψεων ειδήσεων και σε χρήστες με κανονικούς υπολογιστές αφού το μόνο που αλλάζει σε αυτή την περίπτωση είναι οι δυνατότητες της οθόνης που έχει στη διάθεσή του ο τελικός χρήστης (μεγαλύτερη ανάλυση). Η πληροφορία επομένως που καθορίζει το πλήθος των προτάσεων που παρουσιάζονται στον τελικό χρήστη, είτε αυτός κάνει χρήση

συσκευής μικρού μεγέθους, είτε υπολογιστή, είναι η ανάλυση της οθόνης του. Το δεδομένο αυτό αξιοποιείται εσωτερικά από το μηχανισμό μέσα από ένα πίνακα στη βάση δεδομένων που αντιστοιχίζει ανάλυση οθόνης σε πλήθος χαρακτήρων που μπορούν να προβληθούν χωρίς πρόβλημα στην εκάστοτε ανάλυση.

ΚΕΦΑΛΑΙΟ 5

Αλγοριθμικά θέματα και ροή πληροφορίας

> The visionary lies to himself, the liar only to others.
>
> *Friedrich Nietzsche, German philosopher*

Στο παρόν κεφάλαιο δίνεται μια αναλυτική περιγραφή του μηχανισμού που αναπτύχθηκε. Ιδιαίτερη έμφαση δίνεται στη διαδικασία που ακολουθείται από την είσοδο του κειμένου, οι παράμετροι που χρησιμοποιούνται, και όλα τα ενδιάμεσα στάδια μέχρι να φτάσουμε σε μία αποδεδειγμένα καλή περίληψη του κειμένου. Παρουσιάζονται αλγοριθμικά θέματα καθώς και οι διαδικασίες (ροή) που ακολουθείται από το σύστημα.

5.1 Αλγοριθμικά θέματα

Για να αναλύσουμε πως κάθε αλγόριθμος εφαρμόζεται πάνω στα κείμενα, παρουσιάζουμε μια σύνοψη της διαδικασίας εκτέλεσης (Αλγόριθμος 5.1.1).

Αλγόριθμος 5.1.1 Label_article()

```
String Text = fetch_next_text();
List kwfr(text) = create_keyword_frequency_list(text);
List * kwfr_cat(category) = create_keyword_frequency_list(text, category);
Categorize (kwfr(text), *kwfr_cat(category));
if  !Categorize then
   String Stext = Summarize(text,kwfr(text));
   List kwfr(Stext) = create_keyword_frequency_list(Stext);
   List * kwfr_cat(category) = create_keyword_frequency_list(Stext, category);
   Categorize (kwfr(Stext), *kwfr_cat(category));
   if  !Categorize then
      Category = “generic”;
   end if
else
   Summarize(Category,Personal_Data);
end if
```

Παρά το γεγονός ότι η αλγοριθμική διαδικασία δείχνει μόνο την κατηγοριοποίηση των άρθρων, τελικά μέσω αυτής επιτυγχάνουμε τους τρεις βασικούς στόχους: κατηγοριοποίηση, περίληψη και αλληλεπίδραση μεταξύ των μηχανισμών. Ξεκινάμε προσπαθώντας να κατηγοριοποιήσουμε το νέο άρθρο βάσει του συνόλου

εκμάθησης που προυπάρχει στη βάση δεδομένων, δημιουργώντας μια λίστα από αντιπροσωπευτικές κωδικολέξεις (οι οποίες είναι stemmed από την διαδικασία προεπεξεργασίας) μαζί με την συχνότητα εμφάνισής τους. Έπειτα κατασκευάζουμε όμοιες λίστες για όλες τις κατηγορίες που υπάρχουν στη βάση δεδομένων. Αυτές οι λίστες αποτελούνται από τις ίδιες κωδικολέξεις ακολουθούμενες από την συχνότητά τους στην εκάστοτε κατηγορία. Εξετάζουμε την ομοιότητα συνημιτόνου αυτών των λιστών με σκοπό να καθορίσουμε την κατηγορία του κειμένου. Παράδειγμα του αποτελέσματος που προκύπτει φαίνεται στον Πίνακα 5.1.

Κωδικολέξη	**Συχνότητα**
business	0,742862
entertainment	0,449297
health	0,532352
politics	0,418447
science	0,526925
sports	0,642862
education	0,596509

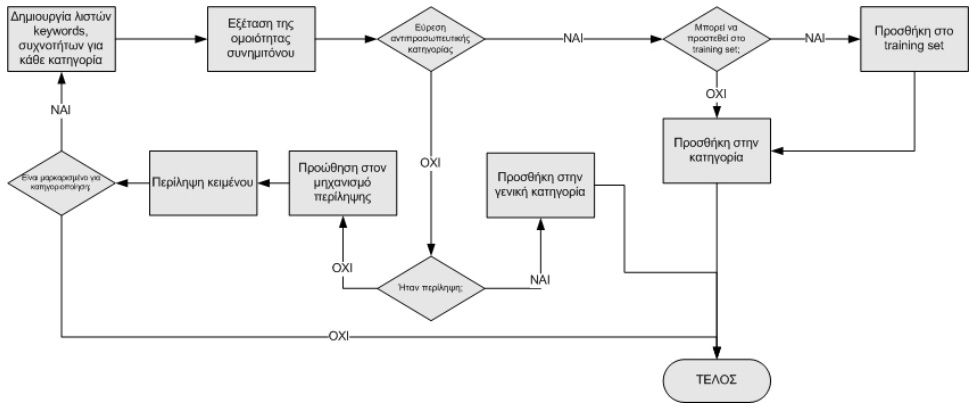
Πίνακας 5.1: Ομοιότητα μεταξύ κειμένου και κατηγορίας

Εάν το κείμενο δεν μπορεί να ταξινομηθεί σε κάποια από τις υπάρχουσες κατηγορίες, τότε προωθείται στον μηχανισμό περίληψης όπου εξάγεται μια γενική (generic) περίληψη η οποία στη συνέχεια εξετάζεται αν μπορεί να κατηγοριοποιηθεί. Ένα κείμενο μπορεί να κατηγοριοποιηθεί επιτυχώς όταν:

- η ομοιότητα συνημιτόνου με κάποια κατηγορία είναι πάνω από ένα όριο, και
- η διαφορά των ομοιοτήτων συνημιτόνου μεταξύ της ισχυρότερης και των υπολοίπων κατηγοριών είναι πάνω από ένα όριο.

Τα όρια αυτά εξηγούνται αναλυτικά στη συνέχεια.

Τελικά, εάν η ομοιότητα συνημιτόνου μεταξύ του κειμένου και της αντιπροσωπευτικής του κατηγορίας είναι πολύ μεγάλη, και παρόμοια η διαφορά των ομοιοτήτων συνημιτόνου μεταξύ της ισχυρότερης και των υπολοίπων κατηγοριών είναι επίσης πολύ μεγάλη, τότε το κείμενο προστίθεται στο δυναμικό σύνολο κειμένων εκπαίδευσης που χρησιμοποιεί ο μηχανισμός. Η προηγούμενη διαδικασία αποτυπώνεται και στο διάγραμμα ροής του Σχήματος 5.1.



Σχήμα 5.1: Το διάγραμμα ροής των διεργασιών του συστήματος.

5.2 Διαδικασίες του συστήματος

Στο Σχήμα 5.1 φαίνεται γενικά η ροή πληροφορίας στο σύστημα που υλοποιήθηκε. Στη συνέχεια ακολουθεί μια πιο λεπτομερής προσέγγιση των διαδικασιών του μηχανισμού.

5.2.1 Προεπεξεργασία κειμένου

Το υποσύστημα προεπεξεργασίας κειμένου και εξαγωγής κωδικολέξεων αποτελεί ένα ανεξάρτητο μηχανισμό που φέρνει εις πέρας μια σημαντική διεργασία του όλου συστήματος, καθώς τροφοδοτεί τους μηχανισμούς που ακολουθούν με την απαραίτητη είσοδο. Πρόκειται για μια αλγοριθμική, ακολουθιακή διαδικασία η οποία περιγράφεται από τον Αλγόριθμο 5.2.1.

Αλγόριθμος 5.2.1 create_keyword_frequency_list(XML,options)

```
String Text = fetch_next_text(XML);
String Title = fetch_next_title(XML);
parseTitle(Title);
Text = removePunctuation(Text);
Text = removeStopwords(Text);
list Keywords = keepKeywordsPercentage(Text);
Keywords = stemming(Keywords);
list keyword_frequency_list = measure_keyword_frequencies(Text,Keywords);
list keyword_positions = get_keywords_positions(Text,Keywords);
return keyword_frequency_list;
```

Η βασική συνάρτηση measure_keyword_frequencies(Text,Keywords) περιγράφεται από τον Αλγόριθμο 5.2.2

Αλγόριθμος 5.2.2 measure_keyword_frequencies(Text,Keywords)

```
for all kw in Keywords do
  if kw is found in Text then
    keyword_frequency_list[kw][appearances]++;
  end if
end for
return keyword_frequency_list;
```

και η get_keywords_positions(Text,Keywords) παρουσιάζεται στον Αλγόριθμο 5.2.3

Αλγόριθμος 5.2.3 get_keywords_positions(Text,Keywords)

```
for all kw in Keywords do
  for all sentence in Text do
    if kw is found in sentence then
      keyword_positions_list[kw][positions].push_back(position);
    end if
  end for
end for
return keyword_positions_list;
```

Οι προηγούμενοι αλγόριθμοι επιτυγχάνουν το ζητούμενο της διαδικασίας του keyword extraction: την εξαγωγή των keywords από το κείμενο, την καταγραφή των συχνοτήτων εμφάνισής τους στο κείμενο και την καταγραφή των προτάσεων στις οποίες εμφανίζονται (θέσεις στο κείμενο). Για να συμβεί αυτό, το κείμενο περνάει από ορισμένα στάδια προεπεξεργασίας, όπως η αφαίρεση των σημείων στίξης, των stopwords καθώς και το stemming του κειμένου. Για την εφαρμοσιμότητα του αλγορίθμου σε πραγματικές συνθήκες λειτουργίας του μηχανισμού, όπου τα άρθρα καταφθάνουν με γοργούς ρυθμούς και η προεπεξεργασία δεν θα πρέπει να διαρκεί πολύ, είναι σημαντικό τα διάφορα μέρη του αλγορίθμου να υλοποιούνται με τρόπο βέλτιστο. Η χρήση επομένως τεχνικών που βασίζονται σε κανονικές εκφράσεις (regular expressions) για την εκτεταμένη διαχείριση συμβολοσειρών την οποία κάνει το υποσύστημα προεπεξεργασίας κειμένου είναι επιβεβλημένη.

5.2.2 Μηχανισμός περίληψης

Περιγραφή

Η διαδικασία παραγωγής περίληψης βασίζεται σε ευρετικές μεθόδους. Αυτό σημαίνει ότι η περίληψη δεν παράγεται 'από την αρχή', αλλά αποτελείται από τις πιο αντιπροσωπευτικές προτάσεις του κειμένου. Με αυτό εννοούμε ότι σε κάθε πρόταση δίνεται ένα 'σκορ' το οποίο μας οδηγεί στην κατασκευή της περίληψης.

Για την παραγωγή της περίληψης ενός άρθρου, 6 ξεχωριστοί παράγοντες χρησιμοποιούνται για την δημιουργία της αλλά και για την αλληλεπίδραση με τον μηχανισμό κατηγοριοποίησης:

(α′) η συχνότητα του keyword στο κείμενο (πόσες φορές εμφανίζεται το keyword στο κείμενο)

(β′) η συχνότητα εμφάνισης του keyword στον τίτλο του κειμένου

(γ′) το ποσοστό των keywords μέσα στην πρόταση

(δ′) το ποσοστό των keywords στο κείμενο

(ε′) η ικανότητα του κάθε keyword να αναπαραστήσει μια κατηγορία, και

(ϛ′) η ικανότητα του κάθε keyword να αναπαραστήσει τις επιλογές και τις επιθυμίες του κάθε ξεχωριστού χρήστη ή μιας κατηγορίας χρηστών με ίδιο προφίλ.

Σύμφωνα με τους δύο πρώτους παράγοντες [(α′) και (β′)], παράγουμε την πρώτη και αρχική εξίσωση για μια γενική βαθμολόγηση των προτάσεων:

$$S_i = \sum w_{k,i}(k_1 + k_2) \tag{5.2.1}$$

όπου, $w_{k,i}$ είναι η συχνότητα του k-οστού keyword της πρότασης i, k_1 είναι μια σταθερά που αναπαριστά την επίδραση του παράγοντα (α′), και k_2 είναι μια σταθερά που αναπαριστά την επίδραση του παράγοντα (β′) στην διαδικασία περίληψης.

Ανάλυση

Μέσα από εκτενή πειραματική διαδικασία, καταλήξαμε σε τιμές για τα k_1 και k_2. Το k_1 ορίζεται από την ακόλουθη σχέση:

$$k_1 = 1 + 0.1x \tag{5.2.2}$$

όπου x οι φορές που ένα keyword εμφανίζεται στον τίτλο του κειμένου. Παρόμοια, το k_2 ορίζεται από την ακόλουθη σχέση:

$$k_2 = 1 + 1.2y \tag{5.2.3}$$

όπου y είναι η πιθανότητα το keyword να βρίσκεται n φορές σε μια πρόταση. Θεωρώντας μια πρόταση με μήκος m (m keywords) και το κείμενο με μήκος t, η παράμετρος y βγαίνει από την ακόλουθη σχέση:

$$y = \frac{n}{t}\frac{m}{t} = \frac{nm}{t^2} \tag{5.2.4}$$

Για να κανονικοποιήσουμε τις τιμές που προκύπτουν από την εξίσωση (5.2.1), προτείνουμε την χρήση των παραγόντων (γ′) και (δ′). Η κανονικοποίηση χρειάζεται διότι, οι μεγάλες σε μήκος προτάσεις του κειμένου, τείνουν να βαθμολογούνται υψηλότερα σε σχέση με τις μικρές σε μήκος. Ο παράγοντας (γ′) αναπαριστά το ποσοστό των keywords στο κείμενο. Πιο συγκεκριμένα, εάν για παράδειγμα τρία keywords έχουν εξαχθεί από μια πρόταση η οποία αποτελείται από πέντε keywords και ο αριθμός των συνολικά εξαχθέντων keywords από το κείμενο είναι είκοσι πέντε, τότε ο παράγοντας (γ′) ισούται με τρία πέμπτα (3/5) και ο παράγοντας (δ′) με τρία εικοστά πέμπτα (3/25).

Η κανονικοποίηση που αναφέρθηκε χρησιμοποιείται για να επιλυθούν κάποια προβλήματα που εγείρονται, όπως στο παράδειγμα που ακολουθεί. Υποθέτουμε ότι ένα κείμενο έχει πολλές μικρές προτάσεις και μία η οποία είναι πολύ μεγάλη. Η μεγάλη πρόταση αποτελείται από 20 keywords και τα keywords που εξήχθησαν (χρήσιμα) είναι 5. Μια μικρή πρόταση, η οποία είναι πολύ αντιπροσωπευτική για το κείμενο αποτελείται από 4

keywords, όλα από τα οποία είναι χρήσιμα. Έστω επίσης ότι ο συνολικός αριθμός των εξαχθέντων keywords για το κείμενο είναι 30. Η μεγάλη πρόταση είναι πολύ πιθανό να βαθμολογηθεί υψηλότερα σύμφωνα με την εξίσωση (5.2.1), αφού το μήκος της την 'βοηθά' να έχει περισσότερα keywords. Οι δύο παράγοντες που προτείνονται, κανονικοποιούν αυτή την πιθανή 'αδικία'. Η μεγάλη πρόταση θα έχει 5/20 και 5/30 αντίστοιχα, ενώ η μικρή πρόταση θα έχει 4/4 και 4/30 για τους παράγοντες (γ΄) και (δ΄) αντίστοιχα. Με αυτό τον τρόπο, η μικρή σε μήκος πρόταση θα αντιμετωπιστεί ως πιο σημαντική σε σχέση με την μεγάλη, κάτι που ισχύει για το συγκεκριμένο κείμενο. Η κανονικοποίηση εφαρμόζεται απ' ευθείας στην εξίσωση (5.2.1) και το $S'_i = S_i N$, όπου το N είναι ο παράγοντας κανονικοποίησης που ισούται με το γινόμενο των (γ΄) και (δ΄).

Οι παράγοντες (ε΄), η ικανότητα του keyword να αντιπροσωπεύει την κατηγορία, και (στ΄), η ικανότητα του keyword να ανταποκρίνεται στις επιλογές του μοναδικού χρήστη, παρουσιάζονται αναλυτικά στις ενότητες που ακολουθούν αφού η επίδρασή τους στην διαδικασία είναι σημαντική και μετατρέπουν το σύστημα εξαγωγής περίληψης σε ένα πλήρως προσωποποιημένο μηχανισμό.

5.2.3 Μηχανισμός κατηγοριοποίησης

Περιγραφή

Το υποσύστημα κατηγοριοποίησης βασίζεται στην μετρική ομοιότητας συνημιτόνου, σε εσωτερικά γινόμενα πινάκων και σε υπολογισμούς ζυγίσματος βαρών. Πιο συγκεκριμένα, το σύστημα αρχικοποιείται με ένα σύνολο κειμένων (άρθρα ειδήσεων) εκμάθησης τα οποία συλλέγονται από σημαντικές ειδησεογραφικές ιστοσελίδες (major news portals). Τα κείμενα αυτά είναι προ-κατηγοριοποιημένα από ανθρώπους και παρουσιάζονται ως ήδη κατηγοριοποιημένα στα news portals. Το σύνολο κειμένων εκπαίδευσης αποτελείται από αυτά τα προκατηγοριοποιημένα κείμενα και από κείμενα που προσθέτονται δυναμικά από τον μηχανισμό όταν εντοπίζονται κείμενα με μεγάλη σχετικότητα με κάποια από τις υπάρχουσες κατηγορίες. Το σύστημα κατηγοριοποίησης δέχεται ως είσοδο την εξαγωγή του μηχανισμού προεπεξεργασίας. Αυτή είναι (α) ένα XML αρχείο (ή δομή) που περιέχει stemmed keywords, την απόλυτη και σχετική συχνότητα εμφάνισής τους αλλά και την θέση τους στο κείμενο και (β) ένα XML αρχείο που περιέχει το ίδιο το κείμενο. Η πληροφορία που αποθηκεύεται στο δεύτερο αρχείο XML αφορά στο id στον τύπο, στον τίτλο και στο σώμα του κειμένου.

Ύστερα από την αρχικοποίηση του συνόλου κειμένων εκπαίδευσης, ο μηχανισμός της κατηγοριοποίησης δημιουργεί λίστες από keywords τα οποία είναι αντιπροσωπευτικά της κάθε μία κατηγορίας, αποτελούμενες από keywords με υψηλή συχνότητα εμφάνισης σε μια συγκεκριμένη κατηγορία και μικρή ή μηδενική εμφάνιση για τις άλλες κατηγορίες. Η δημιουργία των λιστών είναι βοηθητική για την κατηγοριοποίηση των νεοεισερχομένων άρθρων αλλά αποδεικνύεται βοηθητική και για την διαδικασία της εξαγωγής περίληψης.

Ανάλυση

Αφού η διαδικασία περίληψης κειμένου του συστήματος βασίζεται στην επιλογή των πιο αντιπροσωπευτικών προτάσεων οι οποίες επιλέγονται ζυγίζοντάς τες κατάλληλα, τα αποτελέσματα της κατηγοριοποίησης μπορούν να βοηθήσουν στην επιλογή πιο αποτελεσματικού ζυγίσματος για τις προτάσεις. Η κοινή λογική λέει ότι ένα keyword που έχει πολύ υψηλή συχνότητα εμφάνισης για μια συγκεκριμένη κατηγορία, πρέπει να δίνει περισσότερο βάθος σε μια πρόταση που εμφανίζεται, ενώ ένα keyword που έχει μικρή ή μηδενική συχνότητα εμφάνισης για μια κατηγορία μπορεί να προσθέτει λιγότερο στο συνολικό σκορ της πρότασης. Ακόμα παραπέρα, ένα keyword που συμπεριλαμβάνεται στα εξαγόμενα keywords ενός άρθρου που είναι αντιπροσωπευτικό για μια κατηγορία διαφορετική από αυτή στην οποία ανήκει το άρθρο, μπορεί να δώσει αρνητικό βάρος σε μια πρόταση. Η εξίσωση (5.2.5) χρησιμοποιείται για τον υπολογισμό της επίδρασης της διαδικασίας της κατηγοριοποίησης σε αυτήν της περίληψης.

$$k_3 = \begin{cases} A \cdot cw_i & \text{όπου } A > 1 \text{ και } cw \text{ το βάρος κατηγορίας} \\ -A \cdot cw_i & \text{όπου } A > 1 \text{ και } cw \text{ το βάρος κατηγορίας} \\ 1 & \text{για ουδέτερα ή μη βαθμολογημένα από το σύστημα keywords ή εάν } A = 0 \end{cases} \tag{5.2.5}$$

Η παράμετρος A πρέπει να είναι μεγαλύτερη από το 1 και χρησιμοποιείται για να προσθέσει βάρος για την παράμετρο k_3. Εάν θέλουμε η διαδικασία περίληψης να βασίζεται κυρίως στο k_3, τότε οι τιμές ζυγίσματος για το A χρησιμοποιούνται, αντίθετα, αν η διαδικασία περίληψης πρέπει να βασίζεται ισοδύναμα σε όλες τις

'k' μεταβλητές, τότε το A δεν πρέπει να είναι μεγαλύτερο από τις τιμές που έχουν ανατεθεί στα k_1 και k_2. Η παράμετρος cw αποτυπώνει την σχετική συχνότητα ενός keyword στην κατηγορία. Η ποσότητα αυτή μπορεί να μας παρέχει πληροφορία για το πόσο σημαντικό (αντιπροσωπευτικό) είναι ένα keyword για την κατηγορία.

Με την χρήση της εξίσωσης (5.2.5), η εξίσωση (5.2.1) γίνεται:

$$S_i = \sum w_{k,i}(k_1 + k_2)k_3 \tag{5.2.6}$$

5.2.4 Μηχανισμός προσωποποίησης

Περιγραφή

Ο μηχανισμός προσωποποίησης του συστήματος, που υποστηρίζεται ως ένα μέσο επικοινωνίας μεταξύ όλων των διαδικασιών και των χρηστών, μπορεί να χρησιμοποιηθεί για να προσωποποιηθεί η περίληψη σε κάθε χρήστη. Σε ένα σύγχρονο, αποτελεσματικό και χρήσιμο σύστημα, ο χρήστης θα πρέπει να βλέπει προσωποποιημένο περιεχόμενο ανάλογα με τα κριτήρια που έχει θέσει και τις προτιμήσεις του. Στην περίπτωσή μας, θα πρέπει να λαμβάνει προσωποποιημένη περίληψη των άρθρων μόνο που τον ενδιαφέρουν και όχι απλά μιας γενικής μορφής περίληψη που προκύπτει από μια απλή αλγοριθμική διαδικασία.

Σύμφωνα με τις αλγοριθμικές διαδικασίες που ακολουθεί το σύστημα που αναπτύχθηκε, δημιουργούνται λίστες από keywords για κάθε χρήστη οι οποίες αντιπροσωπεύουν τις προτιμήσεις του. Πιο συγκεκριμένα, τα keywords σχηματίζουν δύο ειδών λίστες: μια λίστα 'θετικών' keywords που φαίνεται να ταιριάζουν στις επιλογές του χρήστη (ή της ομάδας χρηστών), και μια λίστα 'αρνητικών' keywords τα οποία δεν ενδιαφέρουν τον χρήστη. Αυτές οι λίστες συνεπάγονται από τις επιλογές των χρηστών για τις κατηγορίες και τα keywords που τον ενδιαφέρουν. Η πρόθεση μας είναι να βαθμολογήσουμε υψηλότερα τις προτάσεις κειμένων που περιέχουν 'θετικά' keywords και χαμηλότερα τις προτάσεις που περιέχουν 'αρνητικά' keywords. Με αυτή την προοπτική, χρησιμοποιείται μια ακόμη παράμετρος, η k_4, η οποία δρα ως παράγοντας προσωποποίησης.

Ανάλυση

Η μεταβλητή για την προσωποποίηση χρησιμοποιείται όπως και αυτή για την κατηγοριοποίηση και δίνεται από την ακόλουθη εξίσωση:

$$k_4 = \begin{cases} B \cdot uw_i & \text{όπου } B > 1 \text{ και } uw \text{ το βάρος χρήστη} \\ -B \cdot uw_i & \text{όπου } B > 1 \text{ και } uw \text{ το βάρος χρήστη} \\ 1 & \text{για ουδέτερα ή μη βαθμολογημένα από τον χρήστη keywords ή εάν } B = 0 \end{cases} \tag{5.2.7}$$

Η παράμετρος uw αποτυπώνει τη σχετική συχνότητα ενός keyword για τον χρήστη. Αυτή μπορεί να μας παρέχει πληροφορία για το πόσο σημαντικό (ισχυρό) είναι ένα keyword για τον χρήστη. Αυτή η παράμετρος προστίθενται στην εξίσωση (5.2.6) η οποία γίνεται:

$$S'_i = \sum w_{k,i}(k_1 + k_2)k_3k_4 \tag{5.2.8}$$

Οι παράμετροι A και B στις εξισώσεις (5.2.5) και (5.2.7) αντίστοιχα, χρησιμοποιούνται σε συνδυασμό μεταξύ τους. Εάν δεν σκοπεύουμε να χρησιμοποιήσουμε κάποιον από τον παράγοντα κατηγοριοποίησης ή προσωποποίησης, μπορούμε να θέσουμε την τιμή 0 για την αντίστοιχη παράμετρο. Εάν θέλουμε να εστιάσουμε την προσοχή μας κυρίως στον παράγοντα προσωποποίησης και λιγότερο στην κατηγοριοποίησης, τότε μπορούμε να θέσουμε $B = 2$ και $A = 1$. Αυτό σημαίνει ότι ο παράγοντας k_4 θα έχει διπλάσια επίδραση από τον k_3. Ο πίνακας 5.2 δείχνει την επίδραση των παραμέτρων (ε') και (στ') σύμφωνα με τις τιμές των A και B.

Όπως παρατηρείται από την εξίσωση 5.2.8, μερικές 'ειδικές' περιπτώσεις μπορούν να λάβουν χώρα από τους μηδενισμούς που εισάγουν οι παράμετροι k_3 και k_4. Ο πίνακας 5.3 δείχνει την αντίδραση του αλγορίθμου στις τέσσερις διαφορετικές καταστάσεις.

Μια ειδική περίπτωση συμβαίνει όταν η μεταβλητή κατηγοριοποίησης είναι αρνητική και η μεταβλητή προσωποποίησης είναι θετική. Σε αυτή την περίπτωση θεωρούμε ότι, η επιλογή του χρήστη για το συγκεκριμένο keyword ως αντιπροσωπευτικό των ενδιαφερόντων του, υπερισχύει της μη αντιπροσωπευτικότητας του keyword για συγκεκριμένη κατηγορία. Επιπρόσθετα, όταν και οι δύο μεταβλητές είναι αρνητικές, το

αποτέλεσμα παραμένει αρνητικό αφού οι αρνήσεις σε αυτή την περίπτωση σημαίνουν ακόμα πιο αρνητικό σκορ για την πρόταση.

5.2.5 Μηχανισμός εφαρμογής σε συσκευές μικρού μεγέθους

Δεδομένου ότι με τις συσκευές μικρού μεγέθους τίθενται θέματα που αφορούν α) στο χαμηλό εύρος ζώνης και β) στη χαμηλή δυνατότητα απεικόνισης (μικρή ανάλυση, μέγεθος οθόνης) και πλοήγησης, είναι φυσικό να επιχειρηθεί μια λύση η οποία θα προσεγγίζει τις δύο αυτές παραμέτρους. Επιθυμούμε δηλαδή, μικρή ποσότητα διακινούμενων δεδομένων και περιεχόμενο με μέγεθος που θα προσαρμόζεται κατάλληλα στην οθόνη της συσκευής του απομακρυσμένου χρήστη. Από τα παραπάνω, δε θα πρέπει να ξεχνούμε και την προσωποποίηση στο χρήστη η οποία σε αυτή την περίπτωση έχει να κάνει με την κατάλληλη μορφοποίηση της απάντησης στις δυνατότητες της συσκευής.

Πίνακας 5.2: Επίδραση των παραμέτρων A και B στο ζύγισμα των προτάσεων

A	**B**	**Αποτέλεσμα**
0	0	Οι παράγοντες προσωποποίησης και κατηγοριοποίησης δε υπολογίζονται στο αποτέλεσμα
0	1	Μόνο ο παράγοντας προσωποποίησης έχει επίδραση στο ζύγισμα των προτάσεων
1	0	Μόνο ο παράγοντας κατηγοριοποίησης έχει επίδραση στο ζύγισμα των προτάσεων
1	2	Ο παράγοντας προσωποποίησης έχει διπλάσια επίδραση σε σχέση με τον παράγοντα κατηγοριοποίησης στο αποτέλεσμα
1	10	Ο παράγοντας προσωποποίησης είναι τόσο μεγαλύτερος από τον παράγοντα κατηγοριοποίησης που η επίδραση του δευτέρου είναι ασήμαντη
1	1	Η ίδια επίδραση και για τον παράγοντα προσωποποίησης και για τον παράγοντα κατηγοριοποίησης
1.2	1.8	Οι τιμές που χρησιμοποιούνται από το μηχανισμό

Πίνακας 5.3: Αντίδραση του αλγορίθμου περίληψης στις μεταβλητές k_3 και k_4

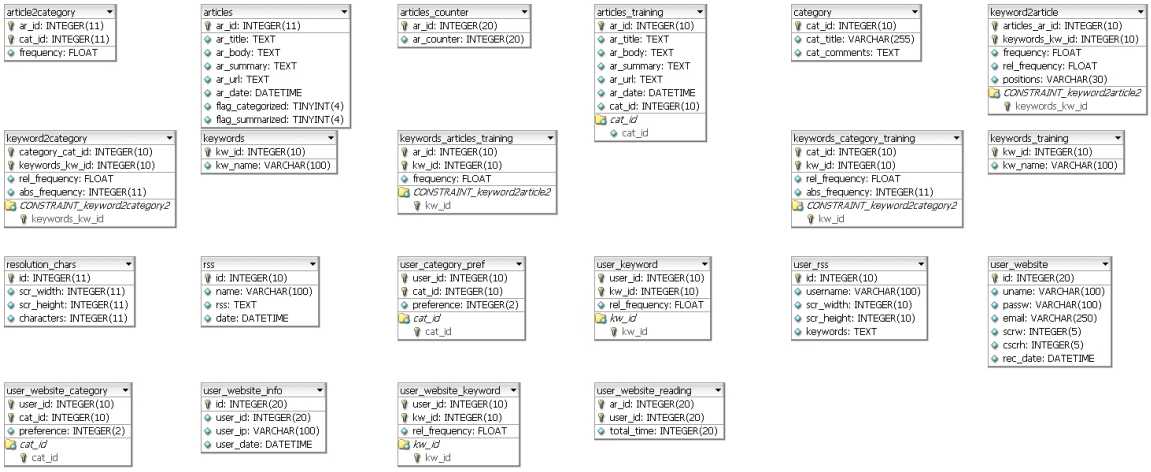
Μεταβλητή k_3	**Μεταβλητή k_4**	**Αποτέλεσμα**
Θετικό	Θετικό	Θετικό
Θετικό	Αρνητικό	Αρνητικό
Αρνητικό	Θετικό	Θετικό (το k_3 δεν συμμετέχει στο αποτέλεσμα)
Αρνητικό	Αρνητικό	Αρνητικό

ΚΕΦΑΛΑΙΟ 6

Βάση δεδομένων του συστήματος

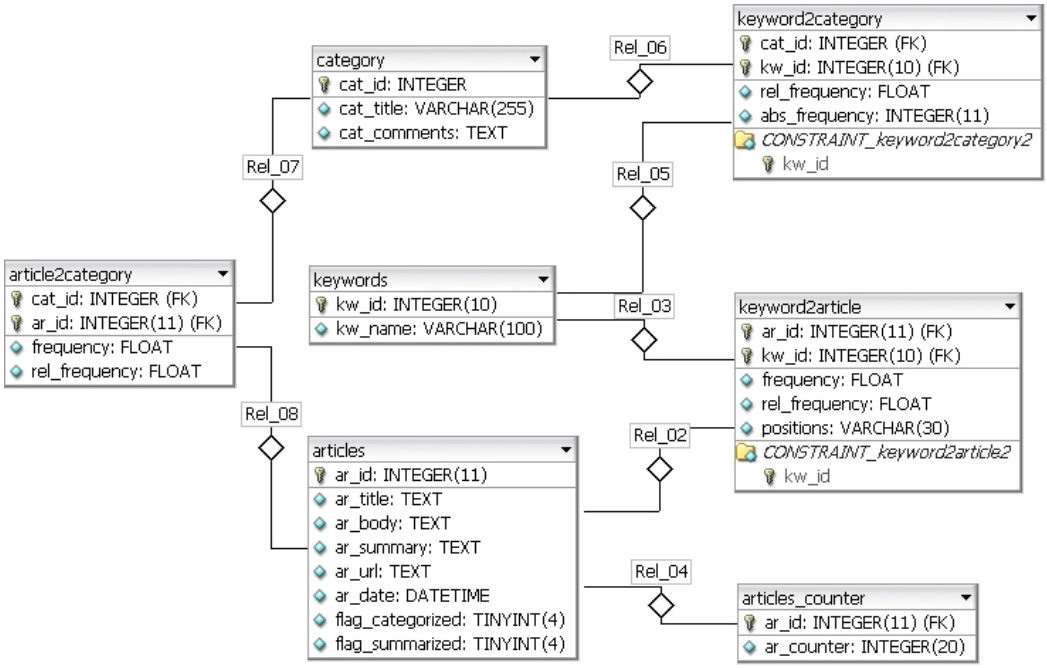
> In mathematics you don't understand things. You just get used to them.
>
> *Johann von Neumann, US (Hungarian-born) computer scientist, mathematician*

Η βάση δεδομένων που χρησιμοποιούμε στο σύστημά μας είναι η έκδοση 5.0.44 της MySQL και η οποία αποτελεί και το ουσιαστικό επίπεδο διασύνδεσης μεταξύ των διαφορετικών υποσυστημάτων που έχουν υλοποιηθεί. Μία γενική εικόνα της βάσης δεδομένων φαίνεται στο σχήμα 6.1.

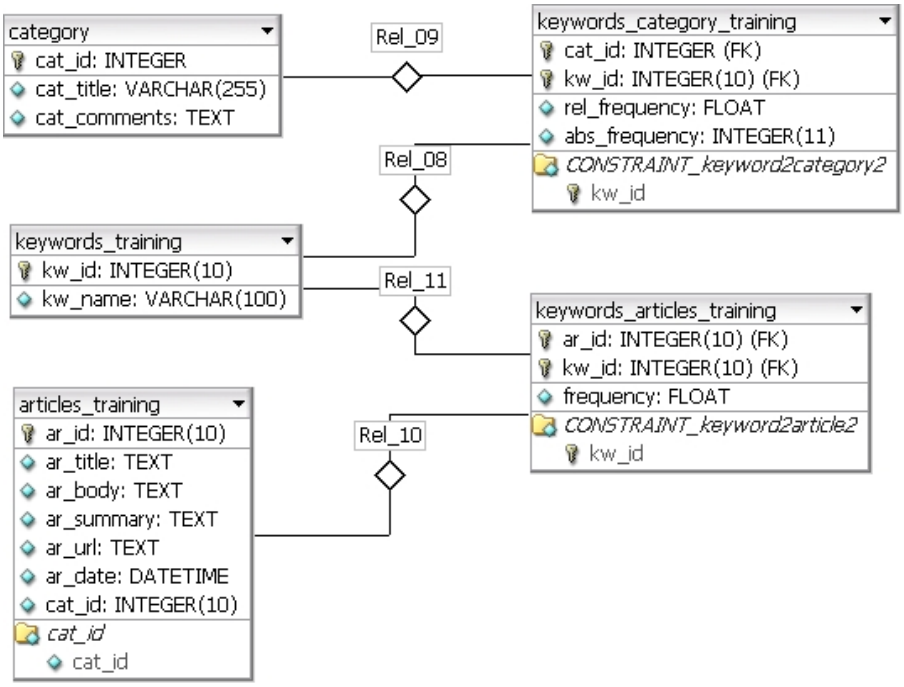


Σχήμα 6.1: Οι πίνακες της βάσης δεδομένων.

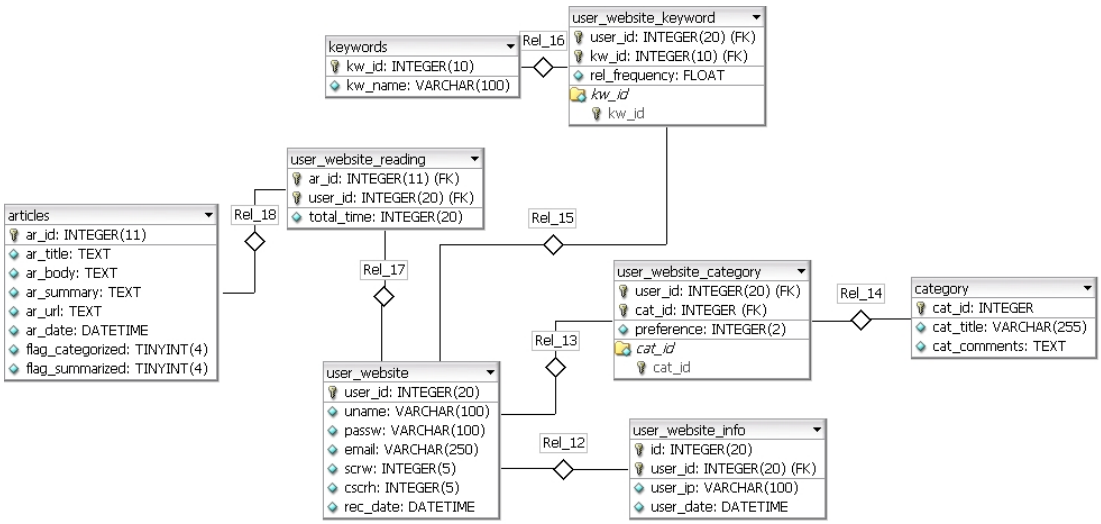
Η εικόνα της βάσης δεδομένων είναι πολύ γενική και οι πίνακές της μπορούν να ομαδοποιηθούν προκειμένου να παρουσιαστεί ο ακριβής τρόπος με τον οποίο γίνεται η αλληλεπίδραση μεταξύ των πινάκων της.



Σχήμα 6.2: Πίνακες που αφορούν τα άρθρα που εισέρχονται στο σύστημα.



Σχήμα 6.3: Πίνακες που αφορούν τη βάση γνώσης του συστήματος.



Σχήμα 6.4: Πίνακες που αφορούν τους χρήστες του συστήματος.

6.1 Ανάλυση γενικών πινάκων

Στο επόμενο κομμάτι ακολουθεί η ανάλυση των πινάκων που υπάρχουν στη βάση δεδομένων και λεπτομερής παρουσίασή τους σε κάθε σημείο χρήσης τους στις διαδικασίες του συστήματος.

6.1.1 *rss*

Ο πίνακας rss χρησιμεύει προκειμένου να παρέχει στον mixed crawler πληροφορίες για το ποιες ιστοσελίδες θα πρέπει να προσπελάσει.

id Μοναδικό αναγνωριστικό πρωτεύον κλειδί για το συγκεκριμένο πίνακα.

name Το όνομα του συγκεκριμένου rss. Καθότι είναι ένα στοιχείο που θα εμφανίζεται στις σελίδες του δικτυακού τόπου θα πρέπει να είναι μικρό και περιγραφικό.

rss Το rss feed από το οποίο ο mixed crawler θα 'διαβάζει' για να εντοπίσει τις καινούριες ειδήσεις που υπάρχουν στους ειδησεογραφικούς δικτυακούς τόπους του συστήματος.

date Η ημερομηνία κατά την οποία προστέθηκε το rss feed.

6.1.2 *articles*

Ο πίνακας articles περιέχει όλα τα στοιχεία που αφορούν τα άρθρα που προστίθενται στο σύστημα.

ar_id Μοναδικό αναγνωριστικό πρωτεύον κλειδί για το συγκεκριμένο πίνακα.

ar_title Ο τίτλος του άρθρου όπως αυτός αναγνωρίστηκε μέσα από τις σελίδες των rss feeds και όχι από την ανάλυση των σελίδων.

ar_body Το κύριο σώμα του κειμένου ή όπως έχει ήδη αναφερθεί το Χρήσιμο Κείμενο.

ar_summary Η γενική περίληψη του άρθρου όπως προκύπτει από το μηχανισμό αυτόματης εξαγωγής περίληψης. Στην περίπτωση της δυναμικής δημιουργίας περίληψης το σύστημα τη συνθέτει σε πραγματικό χρόνο και δεν την ανακτά από τη ΒΔ.

ar_url Το URL το οποίο οδηγεί στη σελίδα του άρθρου όπως αυτό ανακτήθηκε μέσα από το rss feed.

ar_date Η ημερομηνία (timestamp) κατά την οποία ανακτήθηκε ένα άρθρο.

flag_categorized Πρόκειται για μία μεταβλητή αναγνώρισης για να εντοπίσουμε ποια άρθρα έχουν κατηγοριοποιηθεί και ποια όχι προκειμένου ο μηχανισμός κατηγοριοποίησης να είναι σε θέση να αναγνωρίσει ποια άρθρα θα πρέπει να προβούν σε κατηγοριοποίηση.

flag_summarized Πρόκειται για μία μεταβλητή αναγνώρισης για να εντοπίσουμε ποια άρθρα έχουν περάσει από το μηχανισμό αυτόματης εξαγωγής περίληψης και ποια όχι προκειμένου ο μηχανισμός αυτόματης εξαγωγής περίληψης να είναι σε θέση να αναγνωρίσει ποια άρθρα θα πρέπει να προβούν σε διαδικασία αυτόματης εξαγωγής περίληψης.

6.1.3 *keywords*

Πρόκειται για έναν πίνακα που περιέχει όλες τις λέξεις κλειδιά που έχουν καταγραφεί στο μηχανισμό.

kw_id Μοναδικό αναγνωριστικό κλειδί για τις εγγραφές του συγκεκριμένου πίνακα.

kw_name Η λέξη κλειδί (stemmed).

6.1.4 *category*

Ο πίνακας αυτός περιέχει τα στοιχεία των κατηγοριών που υπάρχουν στο σύστημα. Οι κατηγορίες προκύπτουν από τα στοιχεία που διαθέτει η βάση γνώσης του συστήματος.

cat_id Το μοναδικό αναγνωριστικό κλειδί για τις εγγραφές του συγκεκριμένου πίνακα.

cat_name Το όνομα της συγκεκριμένης κατηγορίας.

cat_comments Μικρή περιγραφή για τα στοιχεία κάθε κατηγορίας. Πρόκειται για ένα προαιρετικό πεδίο της ΒΔ που χρησιμεύει για να υπάρχουν μεταδεδομένα εφόσον χρειαστούν μελλοντικά από το σύστημα.

6.1.5 *keyword2article*

Ο πίνακας αυτός χρησιμεύει για να γίνει η συσχέτιση των λέξεων κλειδιών με τα άρθρα του συστήματος. Συγκεκριμένα μας παρουσιάζει ποιες λέξεις κλειδιά υπάρχουν σε κάθε κείμενο του συστήματος.

articles_ar_id Πρόκειται για ένα ξένο κλειδί που αναφέρεται στο μοναδικό αναγνωριστικό κλειδί του άρθρου.

keywords_kw_id Πρόκειται για ένα ξένο κλειδί που αναφέρεται στο μοναδικό αναγνωριστικό κλειδί της λέξης κλειδί.

frequency Η απόλυτη συχνότητα με την οποία εμφανίζεται μία λέξη κλειδί σε ένα κείμενο.

rel_frequency Η σχετική συχνότητα με την οποία εμφανίζεται μία λέξη κλειδί σε ένα κείμενο. Η σχετική συχνότητα υπολογίζεται ως: $rel\_fr_m = \dfrac{abs\_fr_m}{\sum_{k=1}^{L} abs\_fr_k}$

positions Πρόκειται για τις θέσεις μέσα στο κείμενο όπου εντοπίζονται οι λέξεις κλειδιά. Οι θέσεις αφορούν ουσιαστικά τις προτάσεις του κειμένου όπου ευρίσκονται οι λέξεις κλειδιά.

6.1.6 *article2category*

Πρόκειται για έναν πίνακα ο οποίος περιέχει στοιχεία που συσχετίζουν τα άρθρα του συστήματος με κατηγορίες. Κάθε άρθρο δεν αντιστοιχίζεται σε μία μόνο κατηγορία, αλλά το σύστημα μας υπολογίζει τη συσχέτιση με κάθε κατηγορία που υπάρχει στο σύστημα.

ar_id Πρόκειται για ένα ξένο κλειδί που αναφέρεται στο μοναδικό αναγνωριστικό κλειδί του άρθρου.

cat_id Πρόκειται για ένα ξένο κλειδί που αναφέρεται στο μοναδικό αναγνωριστικό κλειδί της κατηγορίας.

frequency Πρόκειται για τη συχνότητα που εκφράζει τη συσχέτιση μεταξύ άρθρου και κατηγορίας. Υπολογίζεται σαν η συσχέτιση του άρθρου με την κατηγορία.

6.1.7 *articles_counter*

Σε αυτό τον πίνακα καταγράφονται τα hits που έχει δεχθεί κάθε άρθρο. Χρησιμοποιείται σαν μετρική που μπορεί να 'δείξει' ποια είναι τα άρθρα για τα οποία δείχνουν ενδιαφέρον οι χρήστες του συστήματος.

ar_id Πρόκειται για ένα ξένο κλειδί που αναφέρεται στο μοναδικό αναγνωριστικό κλειδί του άρθρου.

ar_counter Πρόκειται για έναν ακέραιο αριθμό που καταγράφει τα hits που έχει δεχθεί ένα άρθρο.

6.1.8 *user_website*

Πρόκειται για τον πίνακα που αποθηκεύει τις προσωπικές πληροφορίες κάθε χρήστη.

id Το μοναδικό αναγνωριστικό πρωτεύον κλειδί για τις εγγραφές του συγκεκριμένου πίνακα.

uname Το ψευδώνυμο του χρήστη (username).

passw Ο κωδικός του χρήστη. Για τη βελτιστοποίηση της ασφάλειας του συστήματος ο κωδικός είναι κωδικοποιημένος με md5 κωδικοποίηση. $SHA1$ κωδικοποίηση είναι επίσης δυνατή.

email Το e-mail του χρήστη. Εφόσον ο χρήστης χρησιμοποιεί τη δυνατότητα RSS του δικτυακού τόπου, τότε το e-mail του χρήστη δεν είναι αναγκαία πληροφορία. Εφόσον ο χρήστης χρησιμοποιεί τις υπηρεσίες του δικτυακού τόπου τότε το e-mail μπορεί να βοηθήσει σε κάποιες υπηρεσίες όπως είναι υλοποιημένες στο δικτυακό τόπο.

scrw Πρόκειται για το μήκος της οθόνης του χρήστη σε pixels (screen width). Χρησιμεύει στο να αποσταλεί το σωστό μέγεθος κειμένου στον τελικό χρήστη.

cscrh Πρόκειται για το ύψος της οθόνης του χρήστη σε pixels (screen height). Χρησιμεύει στο να αποσταλεί το σωστό μέγεθος κειμένου στον τελικό χρήστη. Δεδομένου ότι η συνήθης κύλιση σελίδες είναι προς τον κάθετο άξονα (scrolling) το ύψος της οθόνης είναι ενδεικτικό.

rec_date Πρόκειται για την ημερομηνία εγγραφής του χρήστη στο σύστημα (timestamp).

6.1.9 *user_website_category*

Ο πίνακας αυτός αποθηκεύει τις πρωταρχικές επιλογές του χρήστη που αφορούν τις κατηγορίες προτίμησης των χρηστών.

user_id Το μοναδικό ξένο κλειδί που αντιπροσωπεύει τους χρήστες.

cat_id Το μοναδικό ξένο κλειδί που αντιπροσωπεύει τις κατηγορίες.

preference Πρόκειται για την επιλογή του χρήστη user_id όσον αφορά την κατηγορία. *cat_id* Το preference μπορεί να πάρει τιμές από -5 έως 5 με το -5 να αντιπροσωπεύει δυσαρέσκεια προς την κατηγορία και το 5 να αντιπροσωπεύει πλήρη προτίμηση προς την κατηγορία.

6.1.10 *user_website_info*

Ο πίνακας αυτό χρησιμοποιείται σαν log για τις ενέργειες του χρήστη. Καταγράφει τις ημερομηνίες και την IP από την οποία έχουν πραγματοποιήσει σύνδεση οι χρήστες και βοηθά στην καλύτερη παρουσίαση των νέων άρθρων στους χρήστες αφού το σύστημα είναι σε θέση να γνωρίζει ποια άρθρα έχουν προστεθεί στο σύστημα από την τελευταία φορά που το επισκέφθηκαν οι χρήστες του συστήματος.

id Το μοναδικό αναγνωριστικό κλειδί που αφορά τις εγγραφές που γίνονται στο συγκεκριμένο πίνακα. Χρησιμοποιείται γιατί το ξένο κλειδί user_id δε μπορεί να είναι κλειδί στον συγκεκριμένο πίνακα λόγω της πληθώρας των εγγραφών χρήστη που υπάρχουν στο συγκεκριμένο πίνακα και αφορούν μεμονωμένους χρήστες.

user_id Το μοναδικό ξένο κλειδί που αντιπροσωπεύει τους χρήστες.

user_ip Η IP από την οποία έχει συνδεθεί ο χρήστης.

user_date Η ημερομηνία που συνδέθηκε ο χρήστης.

6.1.11 *user_website_keyword*

user_id Το μοναδικό ξένο κλειδί που αντιπροσωπεύει τους χρήστες.

kw_id Το μοναδικό ξένο κλειδί που αντιπροσωπεύει τις λέξεις κλειδιά.

rel_frequency Η σχετική συχνότητα που αντιπροσωπεύει κατά πόσο ο χρήστης ενδιαφέρεται για τη συγκεκριμένη λέξη κλειδί. Οι τιμές είναι θετικές και αρνητικές ενώ συνήθεις τιμές για το συγκεκριμένο πεδίο είναι -2,00 έως 2,00. Το φαινόμενο μία λέξη να ξεφεύγει από αυτά τα όρια είναι (α) ο χρήστης να μην ενδιαφέρεται καθόλου για μία λέξη κλειδί και (β) ο χρήστης να ενδιαφέρεται πολύ για μία λέξη κλειδί όταν οι τιμές είναι μικρότερες του -2 και μεγαλύτερες του +2 αντίστοιχα.

6.1.12 *user_website_reading*

Ο πίνακας χρησιμεύει για να καταμετρήσουμε το χρόνο που κάθε χρήστης «σπαταλά» για να διαβάσει ένα άρθρο. Το χρησιμοποιούμε σαν μετρική προκειμένου να καταμετρήσουμε το ενδιαφέρον του χρήστη για συγκεκριμένα κείμενα.

ar_id Το ξένο κλειδί που αντιπροσωπεύει το άρθρο.

user_id Το ξένο κλειδί που αντιπροσωπεύει το χρήστη.

total_time Ο συνολικός χρόνος που έχει σπαταλήσει ο χρήστης στο συγκεκριμένο άρθρο.

6.2 Πίνακες της βάσης γνώσης

Οι προηγούμενοι πίνακες αντιπροσωπεύουν όλους τους πίνακες που χρησιμοποιούνται άμεσα από το δικτυακό τόπο και από το δυναμικό RSS προκειμένου να παρέχουν και να αποθηκεύουν όλες τις απαραίτητες πληροφορίες. Στην πορεία θα παρουσιάσουμε τους πίνακες που χρησιμοποιούνται για να συνθέσουν τη βάση γνώσης πάνω στην οποία στηρίζονται οι βασικότεροι αλγόριθμοι του συστήματός μας.

6.2.1 *articles_training*

Πίνακας που χρησιμοποιείται για να αποθηκεύσει τη βάση γνώσης πάνω στην οποία στηρίζεται το σύστημα για τους αλγορίθμους κατηγοριοποίησης, αυτόματης εξαγωγής περίληψης και προσωποποίησης στο χρήστη.

ar_id Το μοναδικό αναγνωριστικό κλειδί που αφορά τις εγγραφές που γίνονται στο συγκεκριμένο πίνακα και αφορούν τα άρθρα

ar_title Ο τίτλος του άρθρου.

ar_body Το σώμα του άρθρου.

ar_summary Η περίληψη του συγκεκριμένου άρθρου.

ar_url Ο σύνδεσμος όπου βρίσκεται το άρθρο. Πρόκειται για το σύνδεσμο από τον οποίο «κατέβηκε» ο HTML κώδικας της σελίδας.

ar_date Η ημερομηνία κατά την οποία συλλέχθηκε το άρθρο από τον αυτοματοποιημένο μηχανισμό

cat_id Η κατηγορία στην οποία ανήκει το άρθρο. Ας μην ξεχνάμε πως πρόκειται για τη βάση γνώσης και συνεπώς τα άρθρα είναι προκατηγοριοποιημένα.

6.2.2 *keywords_articles_training*

Ο πίνακας αυτός χρησιμοποιείται προκειμένου να αποθηκευτούν οι λέξεις κλειδιά που έχουν εξαχθεί από τα άρθρα. Για τη βάση γνώσης δεν είναι ένας πίνακας ο οποίος έχει άμεση χρησιμότητα για τους αλγορίθμους του μηχανισμού. Ωστόσο, είναι ένας βοηθητικός πίνακας γιατί χρησιμοποιείται προκειμένου να ελεγχθούν άρθρα τα οποία βρίσκονται στη βάση γνώσης και είναι προβληματικά. Ως προβληματικά αναφέρονται τα άρθρα των οποίων οι λέξεις κλειδιά δεν ανταποκρίνονται στην ενότητα την οποία αντιπροσωπεύει μία κατηγορία.

ar_id Το μοναδικό ξένο κλειδί που αναφέρεται στο άρθρο από το οποίο εξάγουμε τις λέξεις κλειδιά.

kw_id Το μοναδικό ξένο κλειδί που αναφέρεται στις λέξεις κλειδιά που εξάγονται από το άρθρο με αναγνωριστικό κλειδί ar_id. Τα δύο παραπάνω χρησιμοποιούνται ως κλειδιά για το συγκεκριμένο πίνακα και συνεπώς δε μπορεί να υπάρχουν διπλές εγγραφές με τα ίδια χαρακτηριστικά ar_id και kw_id.

frequency Η απόλυτη συχνότητα που αναφέρεται στη συχνότητα με την οποία εμφανίζεται μία λέξη κλειδί σε ένα άρθρο. Είναι μία μετρική η οποία μπορεί να δείξει τη σημαντικότητα μίας λέξης κλειδιού για ένα άρθρο.

6.2.3 *keywords_category_training*

Πρόκειται ίσως για τον πιο σημαντικό πίνακα της βάσης γνώσης. Σε αυτό τον πίνακα αποθηκεύονται πληροφορίες που αφορούν τις λέξεις κλειδιά που αντιπροσωπεύουν μία κατηγορία, ενώ παράλληλα αποθηκεύεται πληροφορία που αφορά το πόσο σημαντική είναι μία λέξη για μία κατηγορία.

cat_id Το μοναδικό ξένο κλειδί που αφορά την κατηγορία στην οποία ανήκει μία λέξη κλειδί.

kw_id Το μοναδικό ξένο κλειδί που αφορά τη λέξη κλειδί που ανήκει σε μία κατηγορία.

rel_frequency Πρόκειται για τη σχετική συχνότητα με την οποία εμφανίζεται μία λέξη κλειδί σε μία κατηγορία.

abs_frequency Η απόλυτη συχνότητα με την οποία εμφανίζεται μία λέξη κλειδί σε μία κατηγορία.

6.2.4 *keywords_training*

Ο πίνακας αυτός χρησιμοποιείται προκειμένου να αποθηκευτούν όλες οι λέξεις κλειδιά που εξάγονται από τα άρθρα της βάσης γνώσης.

kw_id Το μοναδικό αναγνωριστικό κλειδί που αντιπροσωπεύει μία λέξη κλειδί.

kw_name Η λέξη που αφορά το keyword.

6.2.5 *resolution_chars*

Αυτός ο πίνακας χρησιμοποιείται προκειμένου να υπάρχει αποθηκευμένη πληροφορία στη σύστημα για να γνωρίζουμε ανά πάσα στιγμή πόσοι χαρακτήρες πρέπει να εμφανίζονται στις διαφορετικές αναλύσεις που μπορεί να έχει η οθόνη του χρήστη.

id Το μοναδικό αναγνωριστικό κλειδί που αφορά το συγκεκριμένο πίνακα.

scr_width Το μήκος της οθόνης σε pixels (screen width).

scr_height Το ύψος της οθόνης σε pixels (screen height).

characters Οι χαρακτήρες που μπορούν να εμφανίζονται σε οθόνες με το συγκεκριμένο scr_width και scr_height.

ΚΕΦΑΛΑΙΟ 7

Τεχνολογίες Υλοποίησης

I never think of the future - it comes soon enough.

Albert Einstein US
(German-born) physicist

Η επιλογή της τεχνολογίας που θα ακολουθηθεί κατά την κατασκευή ενός σύνθετου συστήματος είναι εξαιρετικά σημαντική προκειμένου να δημιουργηθεί ένα καθολικό σύστημα το οποίο να είναι ευέλικτο, να υποστηρίζει εύκολα αλλαγές και αναβαθμίσεις, να αποτελείται από υποσυστήματα και τέλος να βασίζεται σε ανοιχτά πρότυπα. Το σύστημα που υλοποιήθηκε είναι σύνθετο καθότι έχει βάση το διαδίκτυο αλλά ένα σημαντικό κομμάτι του, ίσως ο πυρήνας, κρύβεται στο μηχανισμό που πραγματοποιεί κατηγοριοποίηση και την περίληψη κειμένου και γενικότερα τη διαχείριση πληροφορίας. Ο τελευταίος μηχανισμός ουσιαστικά δεν έχει καμία επαφή με το διαδίκτυο και φυσικά δεν είναι και απαραίτητο να έχει. Βέβαια, τα δεδομένα που δέχεται προέρχονται από εξόρυξη πληροφορίας στο διαδίκτυο (HTML σελίδες) ενώ τα δεδομένα που εξάγει χρησιμοποιούνται προκειμένου να τροφοδοτήσουν τις συσκευές μικρού μεγέθους των χρηστών.

7.1 Τεχνολογίες Υλοποίησης Μηχανισμού

Οι τεχνολογίες που χρησιμοποιήθηκαν σε κάθε επίπεδο του μηχανισμού είναι διαφορετικές προκειμένου να επιτευχθεί η μέγιστη απόδοση συνολικά του συστήματος με τη χρήση κάθε μίας από αυτές.

7.1.1 Βάση Δεδομένων

Οι πιθανές επιλογές που έχουμε όσον αφορά τη βάση δεδομένων του συστήματος προέρχονται από την επιλογή των τεχνολογιών για τους μηχανισμούς περίληψης και κατηγοριοποίησης κειμένου. Συνεπώς θα πρέπει να επιλεγεί μία βάση δεδομένων η οποία να είναι πλήρως συμβατή με το μηχανισμό που θα κατηγοριοποιεί καθώς επίσης και με τη γλώσσα προγραμματισμού που θα χρησιμοποιηθεί για την κατασκευή της εφαρμογής για συσκευές μικρού μεγέθους.

Γιατί *MySQL*

Η MySQL είναι η δημοφιλέστερη Βάση Δεδομένων ανοιχτού κώδικα που προσφέρεται από το Δίκτυο MySQL. Η αρχιτεκτονική της την κάνει να είναι εξαιρετικά γρήγορη και πολύ εύκολη σε αλλαγές και αναβαθμίσεις. Επιτρέπει επαναχρησιμοποίηση κώδικα όπου αυτό είναι αναγκαίο και παρέχει ένα μινιμαλιστικό τρόπο δημιουργίας στοιχείων διαχείρισης βάσης δεδομένων τέτοιο ώστε να κάνει τη MySQL ασύγκριτη σε ταχύτητα, σε κατάληψη χώρου, σταθερότητα και ευκολία. Ο μοναδικός στο είδος του διαχωρισμός του κεντρικού πυρήνα του server από το μηχανισμό αποθήκευσης κάνει δυνατή την ύπαρξη αυστηρού ελέγχου σε συναλλαγές και μείωση ταχύτητας ή ύπαρξη θεαματικά μεγάλης ταχύτητας με απευθείας προσπέλαση των

δεδομένων, στοιχεία που μπορούν να χρησιμοποιηθούν ανάλογα με τις ανάγκες των χρηστών. Η MySQL περιλαμβάνει αποθήκευση σε μηχανή InnoDB, η οποία υποστηρίζει ασφάλεια στις συναλλαγές και ACID-συμβατή μηχανή αποθήκευσης με commit, rollback, crash recovery και low-level locking δυνατότητες. Η έκδοση της MySQL που βρίσκεται αυτή τη στιγμή σε σταθερή κατάσταση είναι η 5.0.44 και υποστηρίζει πολλά στοιχεία που αφορούν την απόδοση, τη διεθνοποίηση και τη δυνατότητα ένταξης του MySQL server σε άλλα στοιχεία υλικού και λογισμικού. Τα πιο βασικά στοιχεία που χαρακτηρίζουν τη MySQL είναι:

- Υποερωτήματα, που επιτρέπουν στους χρήστες να κάνουν σύνθετα ερωτήματα με μεγάλη ευκολία και αποδοτικά.
- Γρήγορη επικοινωνία μεταξύ server και client μέσα από ένα καινούριο πρωτόκολλο.
- Μικρότερη κατανάλωση πόρων από το server μέσα από βελτιστοποίηση στις βιβλιοθήκες.
- Υποστήριξη Unicode, διεθνείς χαρακτήρες και υποστήριξη αποθήκευσης στην πλειοψηφία των συνόλων χαρακτήρων.
- Υποστήριξη τύπων GIS για ερωτήματα που αφορούν χάρτες και γεωγραφικά δεδομένα.

Τα παραπάνω στοιχεία κάνουν τη MySQL ένα υπερ-πολύτιμο εργαλείο στα χέρια κάποιου χρήστη και τη θέτουν στην 1η θέση για επιλογή ως βάση δεδομένων του συστήματός μας [23]

Γιατί *PostgreSQL*

Η PostgreSQL είναι μια σχεσιακή βάση δεδομένων βασισμένη στα αντικείμενα. Ουσιαστικά προέρχεται από την POSTGRES, V 4.2, που έχει δημιουργηθεί στο πανεπιστήμιο της Καλιφόρνια στο τμήμα Επιστήμης των Υπολογιστών του Μπέρκλεϋ. Μάλιστα το συγκεκριμένα σύστημα υλοποίησε πολλές λειτουργικότητες πολλά χρόνια πριν εφαρμοστούν στα πιο γνωστά από τα σημερινά συστήματα βάσεων δεδομένων.

Η PostgreSQL είναι ένας ανοιχτού κώδικα απόγονος του αρχικού κώδικα που γράφηκε στο Μπέρκλεϋ. Υποστηρίζει SQL92 και SQL99 και προσφέρει πολλά στοιχεία που υποστηρίζουν οι περισσότερες βάσεις δεδομένων τελευταίας τεχνολογίας όπως:

- Σύνθετα ερωτήματα
- Foreign Keys
- Triggers
- Διαφορετικές όψεις
- Ακεραιότητα στις συναλλαγές
- Συνεργασία ταυτόχρονων πολλαπλών εκδόσεων

Επιπρόσθετα, η PostgreSQL μπορεί να εμπλουτιστεί σε στοιχεία από κάποιον έμπειρο χρήστη με πολλούς τρόπους ώστε να υποστηρίζει νέα:

- Τύπους δεδομένων
- Συναρτήσεις
- Διαχειριστές
- Συναθροιστικές συναρτήσεις
- Μεθόδους ευρετηρίου
- Διαδικασιακές γλώσσες

Τέλος, αξίζει να τονιστεί η άδεια χρήσης κάτω από την οποία βρίσκεται η PostgreSQL σύμφωνα με την οποία μπορεί να χρησιμοποιηθεί, αλλαχθεί και διακινηθεί από τον καθένα χωρίς κανένα κόστος [30].

Επιλέγοντας τη Βάση Δεδομένων

Σύμφωνα με τα παραπάνω αλλά και λαμβάνοντας υπόψη μας τους σκοπούς που έχει το σύστημά μας καταλήξαμε στην επιλογή της MySQL σαν τη βάση δεδομένων που θα χρησιμοποιηθεί στο σύστημα. Συγκρίνοντας τις δύο βάσεις δεδομένων μπορούμε να καταλήξουμε στο ότι διαθέτουν πολλά κοινά στοιχεία, ωστόσο η MySQL φαίνεται να είναι πιο γρήγορη στην εξυπηρέτηση των queries και transactions καθώς και πιο διαδεδομένη, λόγοι οι οποίοι την κάνουν πιο ισχυρή. Επιπρόσθετα τα στοιχεία διεθνοποίησης που διαθέτει φαίνονται πολύ χρήσιμα για ένα σύστημα το οποίο μελλοντικά μπορεί να επεκταθεί ώστε να υποστηρίζει πολλές γλώσσες. Ένα άλλο στοιχείο που μας οδηγεί στην επιλογή της MySQL είναι και το γεγονός πως οι βοηθητικοί crawlers που τροφοδοτούν το σύστημά μας με σελίδες HTML υποστηρίζουν βάση δεδομένων MySQL. Τέλος θα πρέπει να λάβουμε υπόψη μας το γεγονός πως δημιουργούμε ένα σύστημα πολυεπίπεδο με τη βάση δεδομένων να είναι ο ουσιαστικός σύνδεσμος μεταξύ των περισσοτέρων κομματιών και συνεπώς μία βάση δεδομένων με μεγάλη σταθερότητα και αξιοπιστία θα προσέδιδε κύρος στο συνολικό σύστημα. Καταλήγουμε λοιπόν στη χρήση Mysql Server έκδοση 5.0.44 [23].

7.1.2 Μηχανισμός περίληψης και κατηγοριοποίησης

Ο μηχανισμός περίληψης είναι ένα σύστημα το οποίο αναλαμβάνει μια πολύ μεγάλη και επίπονη διαδικασία. Προκειμένου να καταλάβουμε τι τεχνολογία πρέπει να χρησιμοποιηθεί θα συνοψίσουμε της εργασίες του μηχανισμού σε μία παράγραφο. Ο μηχανισμός περίληψης δέχεται ως είσοδο αρχεία, ή καλύτερα, δομημένη μορφή XML με στοιχεία για το κείμενο και προχωράει σε μια διαδικασία εξαγωγής λέξεων - κλειδιά για αυτό. Ακολουθεί η διαδικασία αντιστοίχησης λέξεων σε προτάσεις και ακολούθως η βαθμολόγηση των προτάσεων για την εξαγωγή των σημαντικότερων αυτών ώστε να προκύψει η περίληψη του κειμένου. Μια αντίστοιχη διαδικασία ακολουθείται και στην περίπτωση κατηγοριοποίησης ενός κειμένου. Ο μηχανισμός συνεχίζει με ένα επίπεδο προσωποποίησης όπου παράγεται μια προσωποποιημένη περίληψη του κειμένου και αποστέλλεται στο χρήστη (που διαθέτει κινητή συσκευή μικρού μήκους) σε κατάλληλη μορφή (π. χ. RSS Feed). Η επικοινωνία με τη βάση δεδομένων είναι διαρκής σε κάθε φάση του μηχανισμού (αποθήκευση/ανάκτηση keywords και συχνοτήτων, κειμένων, κατηγοριών, στοιχείων προσωποποίησης κ.λπ.).

Είναι φυσική συνέπεια ότι ένας τέτοιος μηχανισμός θα πρέπει να μπορεί να επικοινωνήσει άμεσα και γρήγορα με τη βάση καθώς και να κάνει γρήγορους υπολογισμούς (εσωτερικά γινόμενα, υπολογισμός μέτρων, πράξεις σε πίνακες, κ.λπ.) όπου αυτοί είναι απαραίτητοι. Το ερώτημα που τίθεται εδώ είναι αν θα χρησιμοποιηθεί κάποια αντικειμενοστραφής γλώσσα ή μία γλώσσα διαδικαστική και ποια θα μπορούσε να είναι αυτή.

Γιατί *C*

Η επιλογή της C μπορεί να γίνει για ένα σύνολο από λόγους μεταξύ των οποίων είναι οι εξής: Η C μπορεί να χρησιμοποιηθεί σαν χαμηλού επιπέδου γλώσσα προγραμματισμού επιτρέποντας άμεση πρόσβαση στους πόρους του υπολογιστή και άρα στην αποτελεσματική και χωρίς overhead αξιοποίησή τους. Εξάλλου, είναι η καθιερωμένη γλώσσα για χαμηλού επιπέδου προγραμματισμό που ένας μηχανικός θα απαιτηθεί να κάνει για την καλύτερη αξιοποίηση του υλικού που σχεδιάζει και αναπτύσσει. Ταυτόχρονα, μπορεί να χρησιμοποιηθεί και σαν γλώσσα υψηλού επιπέδου καθώς η πληθώρα των διαθέσιμων βιβλιοθηκών υπερκαλύπτουν τις απαιτήσεις ανάπτυξης λογισμικού επιπέδου εφαρμογής (Application Layer Software). Επίσης είναι σχετικά μικρή και εύκολη στην εκμάθηση, υποστηρίζει top-down και modular σχεδιασμό, υποστηρίζει δομημένο (structured) προγραμματισμό και είναι αποτελεσματική (efficient) αφού παράγει συμπαγή και γρήγορα στην εκτέλεση προγράμματα. Ακόμα είναι φορητή (portable), ευέλικτη (flexible), ισχυρή (powerful), δε βάζει περιορισμούς, γεγονός που συχνά αποβαίνει σε βάρος της και αποτελεί με τη C++ την ευρύτερα χρησιμοποιούμενη γλώσσα σε ερευνητικά και αναπτυξιακά προγράμματα. Επιπλέον η διάθεση του GNU C/C++ Compiler με την GPL άδεια χρήσης κάνει την ανάπτυξη ενός συστήματος με χρήση της γλώσσας C ιδιαίτερα ελκυστική, αφού σε συνδυασμό με το λειτουργικό σύστημα Linux και κάποιο από τα πολλά υπάρχοντα ολοκληρωμένα περιβάλλοντα (IDEs) προγραμματισμού για αυτό, είναι μιας πρώτης τάξεως λύση για το ζήτημα.

Γιατί *C++*

Πρόκειται για την αντικειμενοστραφή εξέλιξη της γλώσσας C. Από το 1998 το C++ Standard αποτελείται από δύο κομμάτια: ο πυρήνας και οι βασικές βιβλιοθήκες. Η τελευταία έκδοση περιέχει βασικές βιβλιοθήκες της C++ και ένα μεγάλο κομμάτι από τις βασικές βιβλιοθήκες της C. Παράλληλα υπάρχουν πολλές βιβλιοθήκες που έχουν συγκεκριμένους σκοπούς και επικεντρώνονται σε συγκεκριμένα στοιχεία και δεν περιλαμβάνονται στις Standard βιβλιοθήκες. Αξιοσημείωτο είναι και το γεγονός ότι είναι σχετικά απλό να ενταχθούν και να χρησιμοποιηθούν βιβλιοθήκες της C μέσα σε προγράμματα γραμμένα σε C++.

Είναι πολύ σημαντικό να γίνει κατανοητό, πως δεν υπάρχει πλέον μία μοναδική γλώσσα που να ονομάζεται C++. Ο όρος αντιπροσωπεύει μία οικογένεια παρόμοιων γλωσσών οι οποίες είναι συχνά υπό- ή υπέρ- σύνολα μεταξύ τους.

Βασικά στοιχεία της C++ περιλαμβάνουν δηλώσεις, function-like casts, inline functions, function overloading, classes, exception handling κ. α. Η C++ συνήθως πραγματοποιεί μεγαλύτερο έλεγχο τύπων σε μεταβλητές απ' ότι η C. Πολλά στοιχεία της C++ τα υιοθέτησε και η C ωστόσο η C99 παρουσίασε πολλά στοιχεία που δεν υιοθετήθηκαν ούτε και υπάρχουν στην C++. Μία πολύ συνηθισμένη πηγή σύγχυσης είναι το ζήτημα ορολογίας: εξαιτίας της παραγωγής από τη C, στη C++ ο όρος αντικείμενο σημαίνει περιοχή μνήμης, όπως και στη C, και όχι ένα class instance, κάτι το οποίο συμβαίνει στις περισσότερες γλώσσες προγραμματισμού.

Η C++ με τις πάμπολλες βιβλιοθήκες που διαθέτει, είτε ανήκουν στην STL, είναι είναι ξεχωριστές (π. χ. boost, mysql++, cgicc, κ. α.), και με τα πλεονεκτήματα ως γλώσσα προγραμματισμού που κληρονομεί από την C, αποτελεί την ιδανική τεχνολογία υλοποίησης για ένα αποτελεσματικό, γρήγορο, real time μηχανισμό, σας αυτό που αναπτύχθηκε στα πλαίσια αυτής της διπλωματικής.

Γιατί *Java*

Η Java αναπτύχθηκε κατ' αρχήν ως γλώσσα για ανάπτυξη ενσωματωμένου λογισμικού (embedded software) και καλύπτει τις αντίστοιχες ανάγκες ενός Μηχανικού συστημάτων. Είναι φορητή, γεγονός που διασφαλίζει τη δυνατότητα εκτέλεσης των Java προγραμμάτων ανεξάρτητα πλατφόρμας υλικού και λογισμικού. Επίσης διαθέτει πολύ μεγάλη βιβλιοθήκη έτοιμων κλάσεων, οι οποίες διευκολύνουν σε μεγάλο βαθμό τη γρήγορη ανάπτυξη αξιόπιστων εφαρμογών και γνωρίζει ραγδαία εξάπλωση σε ερευνητικά και αναπτυξιακά προγράμματα. Ακόμα μπορεί να χρησιμοποιηθεί για προγραμματισμό στο διαδίκτυο και όσον αφορά την υποστήριξη της Αντικειμενοστραφούς Προσέγγισης θεωρείται πιο καθαρή από τη C++ και έτσι θα μπορούσε να θεωρηθεί σαν λογική συνέχεια της C. Τέλος υιοθετεί μεγάλο μέρος της C.

Η Java παρουσιάστηκε σαν μία γλώσσα που είχε αφαιρέσει τα 'βρώμικα' στοιχεία της C++ και είχε εισάγει ένα σύνολο από καλά στοιχεία άλλων γλωσσών όπως η Smalltalk. Η ιστορία της γλώσσας ξεκίνησε όταν μία ομάδα ερευνητών στην προσπάθειά της να αναπτύξει ενσωματωμένο λογισμικό (embedded software) για έξυπνες καταναλωτικές συσκευές στα πλαίσια του project Green, αποφάσισε να αναπτύξει μία νέα γλώσσα μετά τη διαπίστωσή της ότι η C και η C++ δεν ανταποκρίνονται στις απαιτήσεις της. Έτσι τον Αύγουστο του 1991 εμφανίστηκε μία νέα αντικειμενοστραφής γλώσσα με το όνομα OAK, που είναι το ακρωνύμιο του Object Application Kernel. Η γλώσσα απλά προστέθηκε στον κατάλογο των καλών γλωσσών προγραμματισμού με ουσιαστική υποστήριξη σε εφαρμογές τύπου πελάτη-εξυπηρετητή (client-server) και τίποτα παραπάνω.

Μόλις τον Απρίλιο του 1993 έκανε την εμφάνισή του το NCSA MOSAIC 1.0 ως πρώτο γραφικό πρόγραμμα πλοήγησης στο διαδίκτυο (Web browser) και έτσι η γλώσσα άρχισε να κάνει τα πρώτα της βήματα στο χώρο του διαδικτύου με πολύ θετικά αποτελέσματα. Το στοιχείο αυτό ώθησε τη Sun, μετά από μία αποτυχημένη προσπάθειά της να πουλήσει τη γλώσσα (Αύγουστος 93), να χρηματοδοτήσει την ανάπτυξή της για το 1994, αν και το προηγούμενο έτος είχε διακόψει ως μη επιτυχημένο το αντίστοιχο project. Στα μέσα του 1994, αναπτύχθηκε το πρώτο πειραματικό πρόγραμμα πλοήγησης με Java κάτω από το όνομα του WebRunner. Το φθινόπωρο του ίδιου έτους, ο Van Hoff υλοποιεί με Java τον πρώτο Java διερμηνευτή.

Μόλις τον Ιανουάριο του 1995, η γλώσσα πήρε τη σημερινή της ονομασία και εμφανίστηκε η πρώτη επίσημητεκμηρίωσή της με τη μορφή ενός "white paper". Το Μάιο του ίδιου έτους, η Sun παρουσιάζει επίσημα τη Java και το HotJava. Ταυτόχρονα, η Netscape αγόρασε άδεια χρήσης της Java και ενσωμάτωσε τη γλώσσα στη δεύτερη έκδοση του Netscape, του γνωστού προγράμματος πλοήγησης. Στη συνέχεια, ο ένας μετά τον άλλο, οι μεγάλοι κατασκευαστές λογισμικού ανακοίνωσαν την απόφασή τους να χρησιμοποιήσουν

τη Java, με αποκορύφωμα την απόφαση της Microsoft το Δεκέμβρη του 1995. Μία αναλυτική αναφορά στο χρονικό της εξέλιξης της γλώσσας μπορείτε να βρεθεί στο [17].

Γιατί *Perl*

Η Perl είναι μια γενικού σκοπού γλώσσα προγραμματισμού που αρχικά δημιουργήθηκε για την επεξεργασία κειμένου και τώρα χρησιμοποιείται σε μια πλειάδα συστημάτων, συμπεριλαμβανομένων των συστημάτων διαχείριση, ανάπτυξη συστημάτων δικτύου, δικτυακός προγραμματισμούς, ανάπτυξη GUI και άλλα.

Η γλώσσα αυτή σκοπεύει να είναι απλή, αποδοτική και τέλεια παρά 'όμορφη'. Τα κύρια στοιχεία της είναι η ευκολία στη χρήση, η υποστήριξη διαδικασιακού και αντικειμενοστραφή προγραμματισμού και παράλληλα υποστηρίζει πολύ ισχυρούς μηχανισμούς επεξεργασίας κειμένου. Η γενικότερη δομή της προέρχεται κυρίως από τη γλώσσα προγραμματισμού C. Είναι μια διαδικασιακή γλώσσα προγραμματισμού που χρησιμοποιεί μεταβλητές, παραστάσεις, αποδόσεις, μπλοκ κώδικα, συναρτήσεις ελέγχου και υπορουτίνες. Λαμβάνει υπόψη της τον προγραμματισμό σε shell και τα προγράμματα σε perl είναι μεταφραζόμενα. Όλες οι μεταβλητές διαχωρίζονται με ένα συγκεκριμένο χαρακτηριστικό που προηγείται αυτών, επιτρέποντας έτσι καλύτερη σύνταξη. Όπως και το shell του UNIX, η Perl έχει πολλές έτοιμες συναρτήσεις οργανωμένες σε βιβλιοθήκες που αναλαμβάνουν τις περισσότερες απλές εργασίες όπως ταξινόμηση ή διασύνδεση με λειτουργίες του συστήματος.

Η Perl χρησιμοποιεί συσχετιζόμενους πίνακες από το awk και 'κανονικές εκφράσεις' από το sed. Αυτά τα στοιχεία απλοποιούν την ανάλυση λέξεων, τη διαχείριση κειμένου και τη διαχείριση δεδομένων. Στην έκδοση 5 της perl, προστέθηκαν στοιχεία για να υποστηρίζουν σύνθετους τύπους δεδομένων και δομές δεδομένων καθώς επίσης και μοντέλα αντικειμενοστραφούς προγραμματισμού. Σε όλες τις εκδόσεις της perl ο τύπος δεδομένων μίας μεταβλητής βρίσκεται αυτόματα, ενώ αυτόματη είναι και η διαχείριση της μνήμης. Ο μεταφραστής γνωρίζει τον τύπο και τις απαιτήσεις σε αποθηκευτικό χώρο για κάθε τύπο του προγράμματος. Καθορίζει το χώρο που θα καταλαμβάνει κάθε πρόγραμμα και απελευθερώνει πόρους όποτε αυτό είναι εφικτό. Επιτρεπόμενες μετατροπές μεταξύ τύπων γίνονται αυτόματα. Τα παραπάνω βέβαια σημαίνουν ότι δεν επιτρέπονται διαρροές στη μνήμη, σταμάτημα του μεταφραστή ή να διακοπεί η αναπαράσταση των εσωτερικών δεδομένων.

Η χρήση της perl ταιριάζει με τα προβλήματα εύρεσης προτύπου και κανονικών εκφράσεων που αντιμετωπίζονται από τον μηχανισμό προεπεξεργασίας. Τα εργαλεία που χρησιμοποιεί ή perl κάνουν χρήση βέλτιστων αλγορίθμων και η γλώσσα μπορεί να χρησιμοποιηθεί κατά κόρων για εργασίες που έχουν να κάνουν με διαχείριση συμβολοσειρών (string manipulation). Από την άλλη όμως, η χρήση της perl για ανάπτυξη προγραμμάτων οδηγεί σε κώδικα δύσκολα κατανοητό και συντηρήσιμο, ενώ ο κώδικας απαιτεί όχι και τόσο συνηθισμένη σύνταξη.

7.2 Μηχανισμός συλλογής ειδήσεων

Όπως ήδη αναφέρθηκε για το μηχανισμό συλλογής ειδήσεων χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java. Η επιλογή αυτής της γλώσσας για το συγκεκριμένο μηχανισμό είναι γιατί προσφέρει μεγάλη ευελιξία στη διαχείριση πόρων του διαδικτύου αλλά και γιατί διαθέτει APIs ανάλυσης βάση του DOM μοντέλου των σελίδων HTML.

7.3 Μηχανισμός εξαγωγής χρήσιμου κειμένου

Ο μηχανισμός εξαγωγής του χρήσιμου κειμένου είναι ένα επίπεδο πιο κάτω από το μηχανισμό συλλογής ειδήσεων. Πρόκειται για ένα σύστημα το οποίο δεν έχει καμία αλληλεπίδραση με το επίπεδο δικτύου, ούτε και με το επίπεδο χρήστη. Αυτό έχει σαν αποτέλεσμα να πρόκειται για μία διαδικασία που ανήκει σε αυτές χαμηλότερου επιπέδου. Η υλοποίησή της γίνεται αποκλειστικά με C++ καθότι περιέχει πληθώρα διαδικασιών γλωσσολογικής ανάλυσης, ανάλυσης κειμένου, εκτενή χρήση regular expressions και υλοποίηση αλγορίθμων για stemming.

7.4 Μηχανισμός παρουσίασης πληροφορίας και προσωποποίησης

Ο μηχανισμός παρουσίασης πληροφορίας και προσωποποίησης έγινε με χρήση α) PHP σελίδων για την καταγραφή των επισκέψεων σε άρθρα από τους χρήστες του συστήματος (τα links στα οποία δρομολογούνται μέσω του συστήματος) και β) κώδικα σε C++.

Σε αυτό το κομμάτι έχει προκύψει αρκετές φορές το ζήτημα επιλογής τεχνολογίας καθότι μία πιο ολοκληρωμένη πρόταση θα ήταν υλοποίηση όλων των μηχανισμών σε Java και επιλογή JSP με Enterprise Java beans για το δικτυακό κομμάτι. Ωστόσο, η απόκριση της γλώσσας προγραμματισμού Java στις διαδικασίες πυρήνα του συστήματος μας είναι πολύ πιο αργή από τη C++. Αυτό συμβαίνει κυρίως, όπως έχει ήδη αναφερθεί, στην καλύτερη αντιμετώπιση που έχει η C++ όταν εκτελεί διαδικασίες χαμηλού επιπέδου.

7.5 Τεχνολογίες για συσκευές μικρού μεγέθους

Για την τελική παρουσίαση του προσωποποιημένου περιεχομένου στους τελικούς χρήστες που χρησιμοποιούν συσκευές μικρού μεγέθους (PDA's, κινητά τηλέφωνα, κ.ο.κ), έπρεπε να επιλεγεί μια τεχνολογία παρουσίασης που:

- ακολουθεί στα διεθνή πρότυπα
- είναι ευρέως γνωστή και αποδεκτή
- μπορεί να περιγράψει με ακρίβεια και με το βέλτιστο τρόπο την πληροφορία που στέλνεται στο χρήστη
- εφαρμόζεται τόσο σε συσκευές μικρού μεγέθους όσο και σε συνηθισμένους υπολογιστές
- είναι ανεξάρτητη πλατφόρμας και θα απαιτεί το ελάχιστο πρόσθετο κώδικα (εφαρμογές) από τη μεριά του χρήστη
- επιτρέπει την εύκολη και φθηνή αναπαράσταση της πληροφορίας.

7.5.1 Γιατί *XML*

Η XML είναι μια γλώσσα μορφοποίησης (markup language) για κείμενα τα οποία περιέχουν δομημένη πληροφορία. Η δομημένη πληροφορία περιέχει τόσο περιεχόμενο (π. χ. λέξεις, εικόνες, κ.λπ.), όσο και το τι ρόλο παίζει αυτό το περιεχόμενο. Μια γλώσσα μορφοποίησης είναι ένας μηχανισμός για αναγνώριση δομής σε ένα κείμενο. Η XML προδιαγραφή ορίζει έναν στάνταρ τρόπο για προσθήκη μορφοποίησης σε έγγραφα.

Η XML δεν είναι HTML. Στην HTML, τόσο οι ετικέτες tags όσο και τα στοιχεία τους είναι προκαθορισμένα, κάτι τέτοιο δεν ισχύει για την XML όπου οι ετικέτες αλλά και τα στοιχεία τους ορίζονται από τον χρήστη. Η XML δεν καθορίζει ούτε εννοιολογικά δεδομένα ούτε και ένα σύνολο ετικετών. Για την ακρίβεια, είναι μια meta-γλώσσα που χρησιμοποιείται για την περιγραφή γλωσσών μορφοποίησης. Με άλλα λόγια, η XML παρέχει τη δυνατότητα να καθορίζονται tags και πληροφορίες δομής μεταξύ αυτών. Εφόσον δεν υπάρχει κάποιο προκαθορισμένο σύνολο από ετικέτες, δεν υπάρχει και προκαθορισμένη σημασιολογία ετικετών. Όλες οι εννοιολογικές πληροφορίες ενός XML εγγράφου, θα παρέχονται είτε από την εφαρμογή που το επεξεργάζεται, είτε από ξεχωριστά αρχεία που καθορίζουν το στυλ (stylesheets).

Η γλώσσα προγραμματισμού XML περιγράφει μια κατηγορία πληροφοριών (data objects) που καλούνται XML έγγραφα (documents) καθώς επίσης περιγράφει τμηματικά τη συμπεριφορά των προγραμμάτων που τα επεξεργάζονται. Τα XML έγγραφα αποτελούνται από μονάδες αποθήκευσης που καλούνται entities (οντότητες), οι οποίες περιέχουν πληροφορίες αναλυμένες ή μη. Οι αναλυμένες πληροφορίες αποτελούνται από χαρακτήρες (characters) οι οποίοι συνθέτουν character data και άλλοι οι οποίοι συνθέτουν markup. Η μορφή markup κωδικοποιεί την περιγραφή της τελικής αποθήκευσης του εγγράφου καθώς και τη λογική δομή.

Ένα λογισμικό μοντέλο που καλείται επεξεργαστής XML χρησιμοποιείται να διαβάζει XML έγγραφα και παρέχει πρόσβαση στο περιεχόμενο και τη δομή τους. Υποτίθεται ότι ο επεξεργαστής XML λειτουργεί

εκ μέρους ενός άλλου μοντέλου που καλείται application (εφαρμογή). Αυτή η προδιαγραφή περιγράφει την απαιτούμενη συμπεριφορά του επεξεργαστή και συγκεκριμένα πως θα πρέπει να διαβάζει τα XML δεδομένα και ποιες πληροφορίες πρέπει να παρέχει στην εφαρμογή.

Οι προσχεδιασμένοι στόχοι της XML σύμφωνα με το W3C [32] είναι:

1. Η XML πρέπει να είναι εύχρηστη στο Internet.
2. Η XML πρέπει να υποστηρίζει μεγάλη ποικιλία από εφαρμογές.
3. Η XML πρέπει να είναι συμβατή με την SGML.
4. Θα είναι εύκολο να γράφονται προγράμματα που επεξεργάζονται XML έγγραφα.
5. Ο αριθμός των προαιρετικών χαρακτηριστικών στην XML θα είναι όσο το δυνατόν πιο μικρός, ιδανικό επίπεδο το μηδέν.
6. Τα XML έγγραφα θα πρέπει να είναι ευανάγνωστα.
7. Ο σχεδιασμός XML θα πρέπει να προετοιμάζεται γρήγορα.
8. Ο σχεδιασμός XML θα πρέπει να είναι τυπικός και περιεκτικός.
9. Τα XML έγγραφα θα πρέπει να δημιουργούνται εύκολα.
10. Η περιεκτικότητα στον XML συμβολισμό είναι μικρής σημασίας.

7.5.2 Γιατί *RSS*

Το πρότυπο RSS [31], όπως έχει ήδη αναφερθεί, είναι μια οικογένεια από σχήματα τροφοδότησης περιεχομένου στους χρήστες που χρησιμοποιούνται για να δημοσιεύουν συχνά ενημερωμένο περιεχόμενο όπως οι καταχωρήσεις blog, οι τίτλοι ειδήσεων ή τα podcasts. Ένα έγγραφο RSS, που καλείται επίσης και 'feed', περιέχει είτε μια περίληψη του περιεχομένου του σχετικού ιστοχώρου, είτε το πλήρες κείμενο. Το RSS καθιστά δυνατό για τους χρήστες να παρακολουθούν τους αγαπημένους ιστοχώρους τους με έναν αυτοματοποιημένο τρόπο που δεν απαιτεί την πλοήγηση σε αυτούς. Το περιεχόμενο RSS μπορεί να διαβαστεί χρησιμοποιώντας λογισμικό γνωστό και ως 'feed reader' ή 'aggregator'. Από τη στιγμή που ο χρήστης γίνεται συνδρομητής σε ένα feed, ο aggregator αναλαμβάνει να λαμβάνει τα νέα που προέρχονται από το feed ανά τακτά χρονικά διαστήματα. Υπάρχουν διάφορα πρότυπα RSS, RSS 2.0, RSS 1.0, RSS 0.91, όλα όμως χρησιμοποιούν μια XML δομή για τη σύνταξη των δεδομένων.

Το βασικότερο πλεονεκτήματα του προτύπου RSS (όποια έκδοση και αν επιλεγεί), είναι ότι εξοικονομεί χρόνο και εύρος ζώνης στους τελικούς χρήστες, ιδιαίτερα σε αυτούς που χρησιμοποιούν συσκευές μικρού μεγέθους. Η χρήση μάλιστα του προτύπου XML από την τεχνολογία RSS προσδίδει περαιτέρω πλεονεκτήματα στο πρότυπο RSS. Σαν μειονέκτημα του προτύπου RSS θα μπορούσαμε να πούμε ότι είναι η ύπαρξη πολλών 'εκδόσεων' που δημιουργήθηκαν ανά τον χρόνο και ανάλογα με τις απαιτήσεις των εφαρμογών. Η κατάσταση αυτή δημιουργεί ορισμένες ασυμβατότητες στις διάφορες εφαρμογές aggregator, ή τους web browsers που έχουν δυνατότητα απεικόνισης των RSS feeds. Το μειονέκτημα όμως μπορεί να αντιμετωπιστεί χρησιμοποιόντας μόνο τα πρότυπα όπως περιγράφονται από το W3C και εφαρμογές που τα ακολουθούνε πιστά.

7.5.3 Γιατί *CGI*

Τα CGI (Common Gateway Interface) scripts επιτρέπουν να τρέξει ένα εκτελέσιμο πρόγραμμα στον HTTP server. Οι περιπτώσεις στις οποίες χρησιμοποιούνται είναι όταν θέλουμε να επεξεργαστούμε δεδομένα που έρχονται ως αποτελέσματα συμπλήρωσης μιας φόρμας, για τη δημιουργία δυναμικών HTML εγγραφών, για μετρητές προσπελάσεων (counters). Τα CGI scripts μπορούν να γραφούν σε οποιαδήποτε γλώσσα μπορεί να παράγει εκτελέσιμο αρχείο στη μηχανή που τρέχει ο server. Ανάλογα με την πλατφόρμα υλοποίησης , επιλέγεται και η γλώσσα. Έτσι, σε Unix χρησιμοποιούνται PERL, C-shell, C/C++ ενώ σε Windows 95/NT, PERL, Visual Basic, Visual C++.

Ένα CGI script είναι ένα πρόγραμμα το οποίο στο standard output παράγει (συνήθως) HTML ή μορφής XML κώδικα. Σε κάποιες περιπτώσεις παράγει στο standard output κώδικα GIF αρχείου (χρησιμοποιείται στην περίπτωση γραφικών counters σελίδων). Γιάυτό το λόγο αρχικά πρέπει πάντα να τίθεται μια γραμμή προσδιορισμού του περιεχομένου που θα ακολουθήσει. Στο υπόλοιπο μέρος του προγράμματος υπάρχουν εντολές εκτύπωσης του περιεχομένου στο standard output το οποίο και διαβάζει ο Browser ή οποιαδήποτε εφαρμογή επικοινωνεί με τον Web Server (π. χ. RSS Reader). Τα CGI προγράμματα συνήθως αποθηκεύονται σε ένα συγκεκριμένο χώρο. Το directory το οποίο τα περιέχει συνήθως ονομάζεται 'cgi-bin'. Τα αρχεία που αποθηκεύονται εκεί είναι εκτελέσιμα αρχεία που μπορεί να τα τρέξει ένα σύστημα.

Τα CGI scripts αποτελούν μια εύκολη λύση για να προσαρμόσουμε τον υπάρχοντα κώδικα παραδοσιακών γλωσσών προγραμματισμού ώστε να δέχεται και να στέλνει περιεχόμενο μέσω ενός Web Server.

7.5.4 Επιλέγοντας την τεχνολογία για τις συσκευές μικρού μεγέθους

Η χρήση τεχνολογίας XML είναι σαφώς αναγκαία αφού δίνει στο μηχανισμό τη δυνατότητα να περιγράψει εύκολα και με πληρότητα την αποστελλόμενη πληροφορία. Από τη μεριά του χρήστη, η εφαρμογή του θα πρέπει να μπορεί να απεικονίσει σωστά την πληροφορία στην οθόνη της συσκευής του, είτε πρόκειται για κλασική οθόνη υπολογιστή, είτε για οθόνη συσκευής μικρού μεγέθους. Χρειαζόμαστε επομένως μια γενικά αποδεκτή ορολογία για την περιγραφή της εξόδου του μηχανισμού μας που θα καθορίζεται και θα περιγράφεται πλήρως από συγκεκριμένα tags. Για το λόγο αυτό, από την μεριά του προγραμματιστικού κομματιού, ο μηχανισμός μας 'σερβίρει' RSS feeds που ακολουθούν το πρότυπο του RSS 2.0 του W3C στους τελικούς χρήστες. Η αποστολή γίνεται με χρήση CGI scripts τα οποία τοποθετούνται στον C++ κώδικα και αναλαμβάνουν την αποστολή της απάντηση στον χρήστη μέσω ενός Web Browser. Η τεχνολογία CGI μας εξασφαλίζει διαφάνεια και ελάχιστες αλλαγές στον κώδικα του C++ μηχανισμού, ώστε να ανακατευθύνεται η έξοδος από το standard output στην εφαρμογή RSS Reader του τελικού χρήστη.

7.5.5 Διασύνδεση μηχανισμών

Η διασύνδεση των μηχανισμών βασίζεται αποκλειστικά στο επίπεδο βάσης δεδομένων αλλά και στη σειριακή εκτέλεση των διαδικασιών που προσφέρει το λειτουργικό σύστημα. Το γεγονός ότι χρησιμοποιούνται πολλαπλά επίπεδα στην υλοποίηση είναι σωτήριο για ένα τέτοιο σύστημα καθότι υπάρχει ένα επίπεδο το οποίο είναι κοινό για όλα τα υποσυστήματα και συνεπώς είναι εφικτή η ανταλλαγή δεδομένων. Παράλληλα, όλοι οι μηχανισμοί του συστήματος έχουν σχεδιαστεί με τέτοιο τρόπο ώστε να δέχονται δεδομένα από δύο διαφορετικά κανάλια και αντίστοιχα να εξάγουν τα δεδομένα σε δύο διαφορετικά κανάλια, το ένα αυτό της βάσης δεδομένων και το άλλο σε μορφή XML. Μιλούμε για το κλασσικό πρότυπο μίας n-tier αρχιτεκτονικής η οποία επιτυγχάνει διασύνδεση των αυτόνομων μηχανισμών που την αποτελούν στο επίπεδο καναλιού επικοινωνίας. Με αυτό τον τρόπο έχουν μηχανισμούς που αποδεσμεύονται όσο αφορά το κομμάτι της υλοποίησης και δεν έχουν κανένα περιορισμό αρκεί να μπορούν να 'διαβάσουν' δεδομένα από βάση δεδομένων ή από XML αρχεία και αντίστοιχα να είναι σε θέση να 'γράψουν' σε βάση δεδομένων ή σε XML αρχεία.

ΚΕΦΑΛΑΙΟ 8

Ανάπτυξη του συστήματος

> Computer Science is no more about computers than astronomy is about telescopes.
>
> *E. W. Dijkstra*

Μέχρι αυτή τη στιγμή έχουμε αναφερθεί στα βασικά συστήματα που αποτελούν τη βάση για το σύστημά μας αλλά και τους αλγορίθμους που χρησιμοποιούμε προκειμένου να υλοποιήσουμε το κάθε υποσύστημα. Σε αυτό το κεφάλαιο θα εστιάσουμε την προσοχή μας στην υλοποίηση κάθε συστήματος ξεχωριστά. Δεδομένου ότι οι γραμμές κώδικα που γράφηκαν συνολικά για την κατασκευή του συστήματος είναι υπερβολικά πολλές για να παρουσιαστούν, θα εστιάσουμε την προσοχή μας στα πιο σημαντικά σημεία καθώς και στις τεχνικές με τις οποίες υλοποιήθηκε κάθε αλγόριθμος σε κάθε σημείο.

8.1 Υλοποίηση του συστήματος

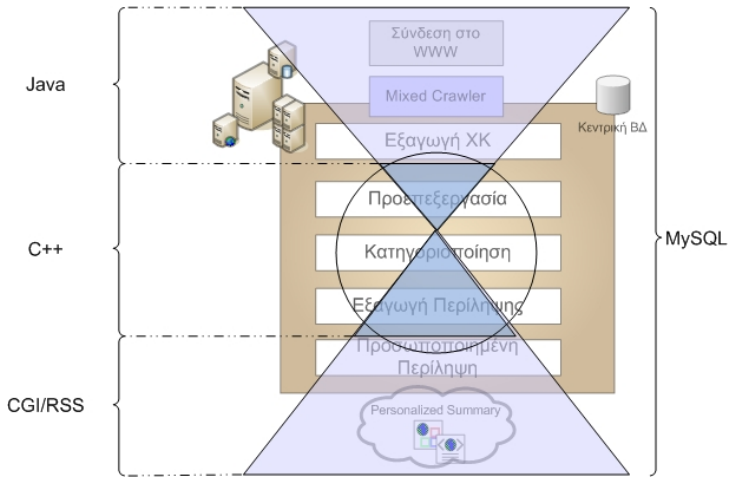
Για την υλοποίηση του συστήματος, όπως έχει ήδη αναφερθεί, χρησιμοποιήθηκε συνδυασμός τεχνολογιών (Σχήμα 8.1). Για τα συστήματα που επικοινωνούν σε υψηλό επίπεδο με τον έξω κόσμο (εκτός μηχανισμού), χρησιμοποιήθηκε η γλώσσα Java και οι τεχνολογίες RSS και CGI, ενώ για τα συστήματα που λειτουργούν στον πυρήνα του συστήματος χρησιμοποιούμε τη γλώσσα προγραμματισμού C++. Για τη διασύνδεση όλων των συστημάτων χρησιμοποιούμε τη βάση δεδομένων η οποία αποτελεί το κανάλι επικοινωνίας όλων των συστημάτων που κατασκευάζουμε, καθότι κάθε σύστημα αντλεί πληροφορίες από αυτή και σε αυτή τις αποθηκεύει.

8.1.1 Συλλογή άρθρων από το διαδίκτυο

Για τη συλλογή των πιο πρόσφατων άρθρων από το διαδίκτυο, υλοποιήθηκε ένας μηχανισμός που θα 'τρέχει' ανά τακτά χρονικά διαστήματα και θα συγκεντρώνει άρθρα από αυτό. Η ιδέα για την υλοποίηση του συγκεκριμένου μηχανισμού στηρίζεται στο γεγονός πως οι μεγαλύτεροι ειδησεογραφικοί δικτυακοί τόποι διαθέτουν RSS feeds τα οποία ανανεώνονται συνέχεια με τις νέες ειδήσεις που προκύπτουν. Η ιδέα είναι να ελέγχονται ανά μία ώρα όλα τα RSS feeds των ειδησεογραφικών πρακτορείων και αν υπάρχουν νέες καταχωρήσεις, τότε ο μηχανισμός να διαβάζει όλα τα νέα άρθρα και να τα προσθέτει στη βάση δεδομένων.

Δεδομένης της μορφής που έχουν τα RSS feeds ο μηχανισμός αυτός μπορεί να συλλέξει την εξής πληροφορία:

- Τίτλος Άρθρου
- URL Άρθρου



Σχήμα 8.1: Τεχνολογίες υλοποίησης του μηχανισμού.

```
<item>
        <title>Taliban extends hostage deadline</title>
        <link>http://edition.cnn.com/2007/WORLD/asiapcf/07/22/afghan.hostages.reut/index.html?eref=edition
        </link>

        <description>Read full story for latest details.
                <a href=http://rss.cnn.com/~a/rss/edition?a=CfB4PD>
                        <img src="http://rss.cnn.com/~a/rss/edition?i=CfB4PD" border="0">
                        </img>
                </a>                                                                    10
        </description>
        <pubDate>Sun, 22 Jul 2007 11:27:25 EDT</pubDate>
</item>
```

Για να εξαχθούν τα συγκεκριμένα στοιχεία χρησιμοποιείται ένας απλός wrapper ο οποίος προσπαθεί να εντοπίσει όλα τα στοιχεία $< title >$ και όλα τα στοιχεία $< link >$. Οι υπόλοιπες πληροφορίες ($< description >$ και $< pubDate >$) δεν είναι σημαντικές για το μηχανισμό αλλά αποθηκεύονται ως μεταδεδομένα.

Η διαδικασία για την εξαγωγή όλων των νέων άρθρων είναι απλή:

- Διάβασμα από τη ΒΔ (πίνακας rss) όλων των RSS που έχουν εισαχθεί στο σύστημα για συλλογή άρθρων
- Ανάλυση όλων των στοιχείων των RSS και προσωρινή αποθήκευσή τους σε μεταβλητή Vector (Java)
- Έλεγχος βάσει URL ή/και τίτλου προκειμένου να εξασφαλιστεί πως το άρθρο δε βρίσκεται ήδη στη βάση δεδομένων
- Εισαγωγή στη ΒΔ (πίνακας articles) όλων των τίτλων URL που δεν υπήρχαν στο σύστημα

Αυτό είναι το ένα κομμάτι του μηχανισμού που συλλέγει όλα τα νέα άρθρα που έχουν εισαχθεί στα ειδησεογραφικά πρακτορεία την τελευταία μία ώρα, δεδομένου ότι ο μηχανισμός εκτελείται κάθε μία ώρα. Από αυτή τη διαδικασία συλλέγουμε όλα τα νέα άρθρα αλλά συγκεκριμένα οι πληροφορίες που έχουμε είναι ο τίτλος του και το URL του. Στο άλλο κομμάτι του μηχανισμού συλλέγουμε όλες τις HTML σελίδες από τα URL που έχει εντοπίσει ο wrapper. Πιο συγκεκριμένα τα βήματα του μηχανισμού είναι τα εξής:

- Ανάγνωση όλων των URL που συνέλεξε ο wrapper
- Σύνδεση με τα URL ένα προς ένα και 'κατέβασμα' της HTML σελίδας
- Αποθήκευση σε μεταβλητή vector όλων των HTML σελίδων που συγκεντρώθηκαν

Αυτό που μας ενδιαφέρει είναι να μπορέσουμε να τροφοδοτήσουμε το μηχανισμό εξαγωγής χρήσιμου κειμένου με τον HTML κώδικα. Φτάνοντας σε αυτό το σημείο του μηχανισμού έχουμε επιτύχει να διαθέτουμε για κάθε νέο άρθρο που εντοπίστηκε τα εξής στοιχεία: Τίτλος, URL, ημερομηνία, μεταδεδομένα, HTML κώδικας.

8.1.2 Εξαγωγή χρήσιμου κειμένου

Η εξαγωγή χρήσιμου κειμένου είναι μία διαδικασία η οποία περιλαμβάνει την απομόνωση των χρήσιμων κομματιών μίας ιστοσελίδας τα οποία στη συγκεκριμένη περίπτωση είναι τα άρθρα - ειδήσεις. Η ανάλυση και εξαγωγή του κειμένου βασίζεται στον τρόπο με τον οποίο είναι δομημένες οι σελίδες που περιέχουν άρθρα - ειδήσεις αλλά και στο DOM μοντέλο στο οποίο μπορεί να αποδομηθεί μία HTML σελίδα.

Ο μηχανισμός εξαγωγής χρήσιμου κειμένου ακολουθεί μετά τη διαδικασία συλλογής άρθρων από το Διαδίκτυο ενώ για μεγαλύτερη ταχύτητα μπορεί να εκτελείται παράλληλα από τη στιγμή που έστω και μία νέα σελίδα συλλέγεται από τους ειδησεογραφικούς δικτυακούς τόπους.

Η εξαγωγή χρήσιμου κειμένου υλοποιείται με τη χρήση της γλώσσας προγραμματισμού Java ενώ παράλληλα έχει ξεκινήσει προσπάθεια μετατροπής του συγκεκριμένου μηχανισμού ούτως ώστε η ανάλυση να γίνεται με C++. Άλλωστε πρόκειται για μία ανάλυση χαμηλού επιπέδου με χρήση πολύπλοκων αλγορίθμων και ως εκ τούτου είναι αναμενόμενη η χρήση της C++ να οδηγήσει σε ακόμα μεγαλύτερες ταχύτητες εκτέλεσης.

Ας περάσουμε όμως στην υλοποίηση του συγκεκριμένου μηχανισμού. Όπως έχουμε ήδη δει (ενότητα 4.3.2), ο HTML κώδικας μπορεί να αναπτυχθεί σε δενδρική μορφή σύμφωνα με το DOM μοντέλο. Αυτό συνεπάγεται πως θα υπάρχουν κόμβοι αλλά και φύλλα. Στη συγκεκριμένη περίπτωση οι κόμβοι αποτελούν τα HTML tags ενώ τα φύλλα περιέχουν το κείμενο που βρίσκεται μέσα στα tags. Τα φύλλα του συγκεκριμένου δέντρου περιέχουν όλο το κείμενο όλης της ιστοσελίδας. Ωστόσο εμείς ενδιαφερόμαστε μόνο για το κομμάτι που περιέχει το άρθρο και όχι για οποιαδήποτε άλλη πληροφορίας η οποία μπορεί να είναι κάποιο άλλο κείμενο της σελίδας ή μενού πλοήγησης. Προκειμένου να πετύχουμε τη σωστή εξαγωγή πληροφορίας κάνουμε μία απλή διαπίστωση. Ο κόμβος πατέρας των φύλλων με χρήσιμο κείμενο έχει τις εξής ιδιότητες:

- Τα φύλλα του παρουσιάζουν μεγάλο ποσοστό σε κείμενο συγκριτικά με όλο το κείμενο που έχει η HTML σελίδα.
- Οι γειτονικοί του κόμβοι έχουν και αυτοί φύλλα με μεγάλο ποσοστό κειμένου συγκριτικά με όλο το κείμενο που έχει η HTML σελίδα.
- Έχουν πολύ περισσότερο κείμενο μέσα σε tags που αφορούν διαμόρφωση κειμένου ($< b >, < i >, < h1 >, < h2 >$, κλπ) παρά σε tags που αφορούν links ($< a >$)

Όπως φαίνεται και από τις ιδιότητες που έχουν τα φύλλα θα πρέπει να ορίσουμε συγκεκριμένες μεταβλητές για να μπορέσουμε να εξάγουμε το χρήσιμο κείμενο. Η μία μεταβλητή που χρειαζόμαστε αφορά το συνολικό κείμενο της σελίδας (μέγεθος κειμένου σε bytes). Η δεύτερη μεταβλητή αφορά το μέγεθος κειμένου κάθε φύλλου (μέγεθος κειμένου σε bytes). Η τρίτη μεταβλητή αφορά το μέγεθος κειμένου φύλλων που αφορά links. Τέλος θα πρέπει να χρησιμοποιηθούν μεταβλητές που θα εκφράζουν τη γειτονικότητα των φύλλων και συνεπώς να χρησιμοποιηθεί ένας αλγόριθμος για την αρίθμηση των κόμβων του δέντρου προκειμένου η αρίθμηση των φύλλων να είναι σειριακή. Έτσι παρά το γεγονός ότι τα φύλλα δεν είναι στο ίδιο βάθος θα πρέπει να ορίσουμε μία μεταβλητή που να αποθηκεύει την αρίθμηση των φύλλων. Επειδή ο αλγόριθμος κατασκευής του δένδρου από την ανάλυση της HTML σελίδας είναι depth first χρησιμοποιούμε έναν επιπλέον μετρητή ο οποίος σηματοδοτεί το κάθε φύλλο και αυξάνεται με την εύρεση νέου φύλλου.

Από τα προαναφερθέντα καταλήγουμε στους παρακάτω παράγοντες:

- S_H = το συνολικό μέγεθος του κειμένου σε bytes. Υπολογίζεται προσθέτοντας όλα τα S_{Lx}.
- S_{Lx} = το μέγεθος κειμένου σε bytes για το φύλλο X. Υπολογίζεται μετρώντας τα bytes αλφαριθμητικών χαρακτήρων σε ένα φύλλο.
- S_{Ax} = το μέγεθος κειμένου του φύλλου X που περιέχεται σε tag $< a >$ (link). Υπολογίζεται μετρώντας τα bytes αλφαριθμητικών μέσα σε tags $< a >$ ενός φύλλου.

- I_X = το αναγνωριστικό κάθε φύλλου σύμφωνα με το μετρητή φύλλων.

Για την αναγνώριση ενός φύλλου σαν φύλλο που περιέχει χρήσιμο κείμενο θα πρέπει να ισχύουν συγκεκριμένες προϋποθέσεις που αφορούν τα ποσοστά κειμένου μέσα σε αυτό συγκριτικά με το συνολικό κείμενο της σελίδας και συγκριτικά με το κείμενο που αφορά συνδέσμους. Έτσι για κάθε φύλλο ελέγχουμε τις ποσότητες:

- $LP = S_{Ax}/S_{Lx}$. Πρόκειται για το Link Percentage το οποίο είναι μία ποσότητα που μας δείχνει πόσο από το κείμενο ενός φύλλου είναι κείμενο που βρίσκεται σε link. Αν αυτή η ποσότητα είναι μεγάλη αυτό σημαίνει πως ο συγκεκριμένος κόμβος είναι ένα navigation menu που η πλειονότητα του κειμένου του βρίσκεται μέσα σε links συνεπώς δε μπορεί να είναι το κείμενο ενός άρθρου το οποίο συνήθως δεν περιέχει πολλά links.

- $TP = S_{Lx}/S_H$. Πρόκειται για το Text Percentage το οποίο είναι μία ποσότητα που μας δείχνει πόσο κείμενο περιέχει ένα φύλλο συγκριτικά με το κείμενο ολόκληρης της σελίδας. Αν αυτή η ποσότητα είναι μεγάλη τότε συνεπάγεται πως το κείμενο αυτού του φύλλου ενδέχεται να είναι 'χρήσιμο κείμενο'.

Αφού απορρίψουμε όλα τα φύλλα με μεγάλο LP και κρατήσουμε όλα τα φύλλα με μεγάλο TP υπολογίζουμε πόσο κοντά (distance) είναι οι κόμβοι με μεγάλο TP. Ο αλγόριθμος είναι απλός και συνίσταται στον υπολογισμό της διαφοράς των τιμών I_X κάθε φύλλου. $D_{X,Y} = I_Y - I_X$.

Τα νούμερα που ορίζουν τα όρια για τα LP, TP και D εξήχθησαν μετά από πειραματικές διαδικασίες σε διάφορους δικτυακούς τόπους που περιείχαν άρθρα και ειδήσεις. Χαρακτηριστικό είναι το παράδειγμα που φαίνεται στο Σχήμα 8.2 για τη λειτουργία του μηχανισμού.

Σχήμα 8.2: Χαρακτηρισμός περιοχών ιστοσελίδας από τον μηχανισμός εξαγωγής χρήσιμου κειμένου.

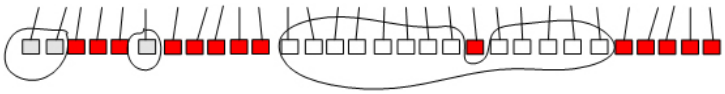
Όπως φαίνεται και από το παραπάνω σχήμα, υπάρχουν περιοχές στο δικτυακό τόπο οι οποίες περιέχουν το κείμενο του άρθρου ενώ άλλες έχουν κείμενο το οποίο δεν αφορά το άρθρο. Οι περιοχές που είναι με κόκκινο χρώμα έχουν αποκλειστεί από χρήσιμο κείμενου λόγω πολύ υψηλού LP. Οι περιοχές με μπλε χρώμα είναι περιοχές που έχουν αποκλειστεί είτε λόγω πολύ χαμηλού TP ή λόγω πολύ ψηλού D. Οι περιοχές με πράσινο χρώμα είναι αυτές που επιλέγονται από το σύστημα σαν το κύριο σώμα του άρθρου.

Ο αλγόριθμος για το σωστό υπολογισμό των παραπάνω περιλαμβάνει τα παρακάτω βήματα:

- Αποδόμηση της HTML σελίδας

- Δημιουργία του DOM μοντέλου με τα tags να αποτελούν κόμβους και τα φύλλα να περιλαμβάνουν μόνο κείμενο.
- Μαρκάρισμα κάθε φύλλου του δένδρου με ένα μοναδικό αναγνωριστικό για το σωστό υπολογισμό της απόστασης.
- Υπολογισμούς των bytes αλφαριθμητικών κάθε φύλλου.
- Μαρκάρισμα του κειμένου που βρίσκεται μέσα σε σύνδεσμο ($< a >$ tag).
- Για κάθε φύλλο
  - Υπολογισμός του LP
  - Αν το LP είναι μεγαλύτερο από 0,42 τότε το κείμενο του φύλλου απορρίπτεται
  - Αν το TP είναι μικρότερο από 0,18 τότε το κείμενο του φύλλου απορρίπτεται
  - Υπολογισμός των D για τα φύλλα που έχουν απομείνει και αν $D > 3$ τότε απόρριψη του κειμένου του φύλλου.

Η επιλογή βάσει γειτνίασης των φύλλων δεν είναι τόσο απλή όσο περιγράφεται παραπάνω. Ουσιαστικά περιλαμβάνει ένα σύνθετο αλγόριθμο που δημιουργεί ομάδες από γειτονικά φύλλα όπως φαίνεται στο Σχήμα 8.3.



Σχήμα 8.3: Χαρακτηρισμός περιοχών ιστοσελίδας από το μηχανισμό εξαγωγής χρήσιμου κειμένου.

Όπως μπορούμε να δούμε, υπάρχουν αρχικά δύο φύλλα τα οποία περιέχουν αρκετό κείμενο ώστε να χαρακτηριστεί χρήσιμο κείμενο αλλά είναι πολύ μακριά από άλλα τέτοια φύλλα. Στη συνέχεια παρουσιάζεται ένα μεμονωμένο και έπειτα μία συστάδα από φύλλα τα οποία έχουν χαρακτηριστεί σαν φύλλα με χρήσιμο κείμενο και τα αποδέχεται ο μηχανισμός. Το συγκεκριμένο παράδειγμα θα μπορούσε να είναι της σελίδες που είδαμε στο παραπάνω σχήμα. Τα πρώτα φύλλα είναι αυτά που περιέχουν τον τίτλο της σελίδας (όχι του άρθρου) ή γενικά στοιχεία που υπάρχουν στη σελίδα ενώ στο σημείο που είναι πολλά φύλλα μαζί βλέπουμε το κυρίως σώμα. Το κόκκινο φύλλο ενδιάμεσα θα μπορούσε να είναι το φύλλο που περιέχει το κείμενο της εικόνας του άρθρου που προφανώς και θέλουμε να απορρίψουμε.

Με αυτό τον τρόπο ο μηχανισμός εξαγωγής χρήσιμου κειμένου είναι σε θέση να μας παρέχει αποκλειστικά και μόνο με χρήσιμο κείμενο που εξάγει από τις σελίδες που έχει ανακτήσει το σύστημα με το μηχανισμό συλλογής άρθρων από το διαδίκτυο.

8.1.3 Προεπεξεργασία κειμένου

Η προεπεξεργασία των κειμένων που δέχεται ο μηχανισμός ως είσοδο, αποτελεί μια βασική και σημαντική διαδικασία του όλου συστήματος, καθώς είναι αυτή που τροφοδοτεί τα συστήματα ανάκτησης πληροφορίας που ακολουθούν με την κατάλληλη είσοδο, η οποία θα πρέπει να είναι σε τέτοια μορφή, ώστε ο μηχανισμός να μπορεί να παράγει ικανοποιητικά αποτελέσματα σαν σύνολο. Αφορά και τη διαδικασία της εξαγωγής κωδικολέξεων (keyword extraction) και πρόκειται ουσιαστικά για μια ακολουθιακή διαδικασία, η οποία μπορεί να θεωρηθεί ως ένα module του όλου συστήματος (και επομένως να αντιμετωπιστεί ξεχωριστά από αυτό).

Το υποσύστημα προεπεξεργασίας δέχεται ως είσοδο ένα πλήθος παραμέτρων:

- Το όνομα του XML αρχείου που περιέχει τα απαραίτητα στοιχεία του κειμένου (τίτλος, σώμα, ID και ενδεχόμενα την κατηγορία του)
- Το ελάχιστο μήκος λέξεων που πρέπει να κρατηθούν
- Ένα σύνολο από λέξεις τερματισμού (stopwords), οι οποίες αφαιρούνται από το κείμενο

- Πληροφορία σχετικά με το ελάχιστο μήκος λέξεων που πρέπει να κρατηθούν και για το αν θα κρατηθούν τα ψηφία (αριθμοί) του κειμένου

Η διαδικασία που ακολουθείται στη συνέχεια περιγράφεται από τα παρακάτω βήματα:

- Parsing του XML αρχείου ώστε να εξαχθούν τα στοιχεία που περιέχει (τίτλος, σώμα κειμένου, είδος (κατηγορία) και αναγνωριστικό (ID»
- Αφαίρεση των σημείων στίξης (punctuation removal) από τον τίτλο του κειμένου και πέρασμα από τον stemmer
- Διαχωρισμός των προτάσεων του κειμένου
- Αφαίρεση των σημείων στίξης του κειμένου
- Αφαίρεση των μεγάλων κενών που υπάρχουν στις προτάσεις του κειμένου. Πλέον κάθε λέξη έχει απόσταση ενός κενού από την επόμενη
- Διαγραφή των stopwords με σύγκριση των λέξεων των προτάσεων με αυτές που έχουν δοθεί ως είσοδος
- Εξαγωγή μεμονωμένων λέξεων από τις προτάσεις (keywords)
- Πέρασμα των keywords του κειμένου από τη διαδικασία του stemming
- Αντιστοίχηση των keywords με τις αρχικές προτάσεις του κειμένου και εύρεση απόλυτης συχνότητας εμφάνισης του κάθε keyword μέσα στο κείμενο
- Κράτημα του ποσοστού των keywords που μας ενδιαφέρει (εξαρτάται από τις διαδικασίες που ακολουθούν το k/w extraction και είναι συνήθως 30-50% των συνολικών keywords)

Η έξοδος που προκύπτει από τη διαδικασία προεπεξεργασίας κειμένου και εξαγωγής keywords που περιγράφηκε είναι:

- Μια λίστα από keywords διατεταγμένη κατά φθίνουσα σειρά συχνότητας εμφάνισης
- Οι σχετικές και απόλυτες συχνότητες εμφάνισης του κάθε keyword μέσα στο κείμενο
- Οι προτάσεις στις οποίες εμφανίζεται το κάθε keyword (π.χ. 1η, 3η, κ.ο.κ)

Οι παραπάνω έξοδοι του μηχανισμού keyword extraction, κωδικοποιούνται κατάλληλα σε αρχείο XML και παρέχονται ως είσοδος στο μηχανισμό που ακολουθεί. Επίσης, είναι δυνατό με κατάλληλο switch στη συνάρτηση που υλοποιεί τη διαδικασία, η έξοδος να αποθηκευθεί απ' ευθείας στη βάση δεδομένων του συστήματος απ' όπου ο μηχανισμός ανάκτησης πληροφορίας που ακολουθεί (περίληψη ή κατηγοριοποίηση κειμένου) να λάβει τις απαραίτητες εισόδους ασύγχρονα.

8.1.4 Κατηγοριοποίηση κειμένου

Η κατηγοριοποίηση κειμένου επιτελεί μια ουσιαστική διαδικασία για το μηχανισμό καθώς μπορεί, δεδομένης μιας βάσης γνώσης κειμένων και ενός συνόλου κατηγοριών, να χαρακτηρίσει ένα νέο κείμενο ταξινομώντας το κατάλληλα με κάποια συσχέτιση στις κατηγορίες. Συνήθως, και εφόσον το κείμενο ανήκει σε κάποια από τις κατηγορίες, η συσχέτιση με αυτή την κατηγορία θα είναι σχετικά μεγάλη, ενώ με τις υπόλοιπες θα είναι πολύ μικρότερη.

Το υποσύστημα κατηγοριοποίησης μπορεί να λειτουργήσει με δύο τρόπους: είτε κατηγοριοποιώντας το κείμενο που δίνεται στην είσοδο, είτε προσθέτοντας το κείμενο της εισόδου στην δυναμική βάση γνώσης (training set) του συστήματος. Φυσικά για την ύστερη περίπτωση, προηγούμενη γνώση για την κατηγορία του κειμένου είναι απαραίτητη. Οι είσοδοι του υποσυστήματος κατηγοριοποίησης κειμένου περιλαμβάνουν τα εξής:

- Το XML αρχείο του keyword extraction που περιέχει τα keywords του κειμένου, τις συχνότητες εμφάνισής τους και τις θέσεις τους στο κείμενο (το τελευταίο στοιχείο δεν χρειάζεται για την κατηγοριοποίηση).
- Ένα training set flag που αντιπροσωπεύει τον τρόπο λειτουργίας από τους δύο που περιγράφηκαν προηγουμένως.
- Το ποσοστό των keywords που θα πρέπει να κρατηθούν από το XML αρχείο των keywords. Αυτό δίνεται ξεχωριστά και ανεξάρτητα από τη διαδικασία του keyword extraction γιατί κρατάμε διαφορετικά μεγέθη του συνόλου των keywords σε κάθε περίπτωση. Οι λόγοι και οι επιλογές οι οποίες γίνονται περιγράφονται αναλυτικά στη συνέχεια.

Ποσοστό των *keywords* για *training set*

Για την περίπτωση που έχουμε να κάνουμε προσθήκη στο training set της βάσης γνώσης μας, κρατούμε ένα ποσοστό 50% των αρχικών keywords λόγω του ότι θέλουμε το κείμενο να προσθέσει τη δικιά του συσχέτιση στην κατηγορία όπου εισάγεται (να μεταβάλει επομένως ελαφρώς την κατηγορία) και επομένως νέα κείμενα που εισέρχονται στο σύστημα και μοιάζουν με αυτό που προστέθηκε στο training set, να κατηγοριοποιούνται στην ίδια κατηγορία. Επίσης η επιλογή του 50% αποκλείει την συμπερίληψη keywords στην κατηγορία τα οποία έχουν πολύ μικρή συχνότητα εμφάνισης στο κείμενο και επομένως δεν είναι τόσο αντιπροσωπευτικά αυτού, άρα και της κατηγορίας.

Ποσοστό των *keywords* για κατηγοριοποίηση

Το ποσοστό των keywords που κρατούνται για την κατηγοριοποίηση ενός νέου άρθρου από το σύστημα είναι 30%. Το αποτέλεσμα αυτό προέκυψε ύστερα από πειραματική διαδικασία η οποία περιγράφεται στην ενότητα 10.1.3 και μας επιτρέπει να έχουμε πολύ καλή συσχέτιση των keywords με το αρχικό κείμενο, χωρίς παράλληλα να χρειάζεται να υπερφορτώνουμε τη βάση δεδομένων με μη χρήσιμα στοιχεία. Η επιλογή αυτή μας δίνει παράλληλα και πλεονέκτημα χρόνου στην εκτέλεση του αλγορίθμου κατηγοριοποίησης, μιας και υπάρχουν λιγότερα keywords τα οποία θα πρέπει να συγκριθούν με σχετικότητες αποθηκευμένες στη βάση δεδομένων.

Διαδικασία κατηγοριοποίησης

Η διαδικασία της κατηγοριοποίησης ενός νέου άρθρου προχωρά ως εξής:

1. Ανακτούνται οι συχνότητες των keyword του κειμένου με κάθε κατηγορία από τη ΒΔ (τα keyword που εμφανίζονται στο κείμενο και υπάρχουν και στην κατηγορία). Έχουμε επομένως, εκτός από τον αρχικό πίνακα συχνοτήτων των keywords του κειμένου, και ένα πίνακα για κάθε κατηγορία που περιέχει συχνότητα εμφάνισης για την κατηγορία του κάθε keyword του κειμένου που εμφανίζεται και στην κατηγορία. Προφανώς, αν κάποιο από τα keywords του κειμένου δεν εμφανίζεται στην εκάστοτε κατηγορία, η αντίστοιχη θέση στον πίνακα συχνοτήτων της κατηγορίας θα έχει την τιμή 0 (μη εμφάνιση).
2. Υπολογίζονται τα μέτρα των προηγούμενων πινάκων, το εσωτερικό τους γινόμενο και από αυτά, η ομοιότητα συνημιτόνου μεταξύ τους. Έχουμε επομένως για κάθε κατηγορία, μια συσχέτιση του κειμένου, κάτι που είναι και το ζητούμενο.
3. Οι συσχετίσεις κειμένου-κατηγορίας ταξινομούνται κατά φθίνουσα σειρά.
4. Πλέον οι συσχετίσεις μπορούν να αποθηκευθούν στη βάση και η κατηγοριοποίηση του κειμένου έχει ολοκληρωθεί.

Διαδικασία προσθήκης στο *training set*

Όπως ήδη αναφέρθηκε, το module της κατηγοριοποίησης είναι εκείνο που διαχειρίζεται τη βάση γνώσης του συστήματος και επομένως έχει τη δυνατότητα προσθήκης νέων άρθρων, αντιπροσωπευτικών των κατηγοριών, σε αυτή. Η διαδικασία που ακολουθείται είναι απλή: έχοντας ως δεδομένα τα keywords του κειμένου (50% των συνολικών) και την κατηγορία την οποία αντιπροσωπεύει το κείμενο, εισάγονται (εάν δεν υπάρχουν ήδη) τα keywords του κειμένου και ανανεώνονται (η εισάγονται) οι συσχετίσεις των keywords με την κατηγορία, στους κατάλληλους training πίνακες. Τέλος εισάγεται και η συσχέτιση των keywords που εισήχθησαν (η τροποποιήθηκαν) με το κείμενο που μόλις μπήκε στο training set. Φυσικά σε κάθε περίπτωση εξασφαλίζεται η μη ύπαρξη διπλοεγγραφών στη βάση και η γενικότερη συνέπεια των δεδομένων. Η διαδικασία ανανέωσης του training set είναι πολύ ευκολότερη αφού δεν εμπεριέχει υπολογισμούς εσωτερικών γινομένων ή ομοιοτήτων συνημιτόνου όπως η διαδικασία της κατηγοριοποίησης, παρά μόνο συναλλαγές με τη ΒΔ.

8.1.5 Αυτόματη εξαγωγή περίληψης

Η διαδικασία της εξαγωγής περίληψης κειμένου, δέχεται για είσοδο την έξοδο του keyword extraction του μηχανισμού, δηλαδή τις εξαγμένες κωδικολέξεις μαζί με τις συχνότητες εμφάνισής τους στο κείμενο καθώς και τις θέσεις τους στις προτάσεις. Επίσης ως είσοδος δίνεται το μέγεθος της απάντησης που επιθυμούμε και αν υπάρχει, πληροφορία για την κατηγορία του κειμένου.

Έχει ήδη αναφερθεί, ότι η διαδικασία αυτόματης εξαγωγής περίληψης δίνει ένα βαθμό, ή αλλιώς ένα σκορ, σε κάθε πρόταση του κειμένου ανάλογα με τη σχετικότητα που εκτιμά πως έχει. Το σκορ της κάθε πρότασης σχηματίζεται με τη βοήθεια ενός πλήθους παραμέτρων που αφορούν στη συχνότητα εμφάνισης του keyword στο κείμενο, στην πιθανότητα εμφάνισης του keyword στον τίτλο του κειμένου, στην κατηγορία στην οποία ανήκει το κείμενο και τέλος στις ιδιαίτερες προτιμήσεις του χρήστη για τις κατηγορίες και επομένως για ορισμένα keywords. Το σύνολο των παραμέτρων συνοψίζεται στη σχέση 5.2.8. Για την υλοποίησή μας, και ύστερα από πειράματα, καταλήξαμε στο να θέσουμε τον μεν παράγοντα $k_1 = 1,4$, ενώ τον παράγοντα $k_2 = 1,2$. Ο πρώτος αφορά στην περίπτωση εμφάνισης του keyword στον τίτλο, ενώ ο δεύτερος στη συχνότητα εμφάνισης του keyword στο κείμενο. Η διαδικασία έχοντας αυτές τις δύο παραμέτρους μπορεί να δώσει μια βασική βαθμολόγηση για τις προτάσεις του κειμένου και επομένως μια περίληψη.

Η ποιότητα της περίληψης αυξάνεται δραματικά με την χρήση των επόμενων δύο ευρετικών.

1. Έχοντας την πληροφορία για την κατηγορία του κειμένου, η διαδικασία αναζητεί για κάθε keyword κάθε πρότασης την σχετικότητα που έχει το keyword με την κατηγορία. Σημειώνοντας το πόσο σχετικό είναι το κάθε keyword ή όχι με την κατηγορία, οι προτάσεις βαθμολογούνται με θετικό βάρος για κάθε keyword σχετικό που περιέχουν και με αρνητικό βάρος για κάθε keyword μη σχετικό που έχουν. Το τελικό σκορ των προτάσεων προκύπτει ύστερα από την προσθαφαίρεση όλων των βαρών. Με αυτό τον τρόπο, οι προτάσεις που περιέχουν keywords αντιπροσωπευτικά της κατηγορίας επιτυγχάνουν υψηλότερο σκορ σε σχέση με άλλες που δεν περιέχουν πολλά αντιπροσωπευτικά keywords και επιτυγχάνουν ουδέτερο σκορ (κοντά στο 0), ή με άλλες που έχουν πολλά αντιπροσωπευτικά άλλων κατηγοριών keywords και επιτυγχάνουν αρνητικό σκορ.

2. Θέλοντας να παράγουμε μια προσωποποιημένη περίληψη για κάποιον χρήστη με δεδομένο προφίλ στο σύστημα, βαθμολογούμε υψηλότερα προτάσεις που περιέχουν keywords αντιπροσωπευτικά των προτιμήσεων του χρήστη (keywords που ανήκουν με υψηλή θετική βαρύτητα στο προφίλ του χρήστη), ή βαθμολογούμε χαμηλότερα ή αρνητικά προτάσεις που περιέχουν keywords μη αντιπροσωπευτικά των προτιμήσεων του χρήστη. Η βαθμολόγηση γίνεται όπως και στην περίπτωση της κατηγοριοποιημένης περίληψης αναζητώντας για κάθε keyword, κάθε πρότασης, τη σημαντικότητα που έχει για τον χρήστη.

Τα προηγούμενα ευρετικά καθορίζουν τα k_3, k_4 της σχέσης 5.2.8 για την κάθε πρόταση. Το k_4 αποφασίστηκε να έχει μεγαλύτερο βάρος από το k_3 κατά έναν λόγο 2 (διπλάσια επίδραση)· δηλαδή το A της σχέσης 5.2.5 τέθηκε ίσο με 1 και το B της σχέσης 5.2.7 τέθηκε ίσο με 2. Το τελικό σκορ κάθε πρότασης προκύπτει συνολικά από τις παραμέτρους k_1, k_2, k_3, k_4, οι βαθμολογίες ταξινομούνται σε φθίνουσα σειρά και υπολογίζεται η απάντησης που πρέπει να επιστρέψει η διαδικασία. Το μέγεθος της εξόδου μπορεί είτε να καθορίζεται ως ποσοστό % επί των προτάσεων του κειμένου εισόδου, είτε ως επιθυμητό πλήθος χαρακτήρων. Οι προτάσεις που τελικά θα αποσταλούν ως περίληψη, ταξινομούνται σύμφωνα με τη σειρά εμφάνισής τους

στο κείμενο, διατηρώντας έτσι την νοηματική συνοχή της απάντησης, και τελικά επιστρέφονται ως έξοδος της διαδικασίας.

8.1.6 Προσωποποίηση περίληψης στο χρήστη

Η προσωποποίηση στο χρήστη είναι ένα από τα πιο σημαντικά κομμάτια του συστήματος καθώς σε αυτό το στάδιο διαμορφώνεται το δυναμικό προφίλ και προβάλλονται πίσω στο χρήστη όλα τα αποτελέσματα των προηγούμενων μηχανισμών.

Η προσωποποίηση στο χρήστη γίνεται σε επίπεδο διαδικτύου με τη συνεργασία PHP, C++ και βάσης δεδομένων. Η προσωποποίηση βασίζεται σε συγκεκριμένες παραμέτρους προκειμένου να είναι πληρέστερη και να είναι εφικτή η καλύτερη δημιουργία προφίλ χρήστη. Οι παράμετροι που θέσαμε στο σύστημα για την προσωποποίηση είναι:

- Οι επιλογές του χρήστη που αφορούν τις κατηγορίες που έχει το σύστημα (μόλις κάνει εγγραφή)
  - Βαθμολόγηση των κατηγοριών ανάλογα με το πόσο ενδιαφέρουν το χρήστη
- Οι επιλογές του χρήστη μόλις του εμφανίζονται άρθρα
  - Επιλογή του χρήστη να διαβάσει ένα άρθρο

Τα παραπάνω αποτελούν παραμέτρους που διαμορφώνουν το προφίλ ενός χρήστη. Όμως ας δούμε τι εννοούμε όταν αναφερόμαστε στο προφίλ ενός χρήστη. Δεδομένων των διαδικασιών με τις οποίες εξάγονται τα αποτελέσματα τόσο για την κατηγοριοποίηση (επιλογή σε ποια κατηγορία ανήκει ένα άρθρο που μόλις μπήκε στο σύστημα) όσο και για τις περιλήψεις έχουμε δει πως αυτό που έχει τη μεγαλύτερη σημασία είναι να εντοπίσουμε τις λέξεις κλειδιά. Έτσι, λοιπόν, και για το προφίλ του χρήστη αυτό που πραγματοποιούμε είναι να δημιουργήσουμε λίστες με λέξεις κλειδιά που έχουν κάποια βάρη. Σε αυτή την περίπτωση τα βάρη είναι θετικά και αρνητικά και προδίδουν το κατά πόσο ο χρήστης ενδιαφέρεται για κάποια λέξη κλειδί ή όχι καθώς και το μέγεθος ενδιαφέροντος.

Ως δεδομένα έχουμε στο σύστημά μας έχουμε 7 κατηγορίες τις οποίες τις χαρακτηρίζουν λέξεις κλειδιά με συγκεκριμένα βάρη. Ο πίνακας 8.1 δείχνει ένα τέτοιο παράδειγμα για μία από τις κατηγορίες του συστήματός μας.

Cat id	Kw id	Rel frequency	Abs frequency
1	42	0.00105974	298
1	43	0.000927275	201
1	44	0.00172208	201
1	41	0.0103325	188
1	37	0.00516625	150
1	228	0.0149689	148
1	45	0.00251689	141

Πίνακας 8.1: Συσχέτιση λέξεων κλειδιών με κατηγορία

Το σκεπτικό είναι πως κάθε χρήστης αντιπροσωπεύει μία υποκατηγορία ή πιο σωστά, μία σειρά από υποκατηγορίες. Αυτό σημαίνει πως εφόσον οι λίστες με τις λέξεις κλειδιά δύνανται να χαρακτηρίσουν μία κατηγορία αυτό συνεπάγεται και πως λίστες με λέξεις κλειδιά δύνανται να χαρακτηρίσουν τις επιλογές και τις προτιμήσεις ενός χρήστης. Αυτό που μας ενδιαφέρει συνεπώς είναι να μπορέσουμε από τις διαδικασίες που περιγράψαμε παραπάνω να καταλήγουμε σε λέξεις κλειδιά και συγκεκριμένα βάρη σε κάθε μία προκειμένου να χαρακτηρίσουμε το χρήστη. Σε πρώτη φάση αυτό που κάνουμε είναι να διαμορφώσουμε κάποιο αρχικό προφίλ για το χρήστη κατά τη διάρκεια που πραγματοποιεί εγγραφή στο σύστημα. Δεδομένου ότι θέλουμε να κρατήσουμε τις διαδικασίες όσο το δυνατόν πιο διαφανείς προς τους χρήστες είναι ίσως το μόνο σημείο που μπορούμε ανώδυνα να βάλουμε το χρήστη στη διαδικασία του να συμπληρώσει κάποια στοιχεία για το προφίλ του.

Η διαδικασία εγγραφής και γενικά το περιβάλλον διεπαφής αποτελούν την βασική μονάδα επικοινωνίας του χρήστη με το σύστημα. Ένας χρήστης εγγράφεται στο σύστημα δίνοντας πληροφορίες για το μέγεθος της συσκευής που χρησιμοποιεί και δίνοντας πληροφορίες για τις κατηγορίες που θέλει να παρακολουθεί.

Ένας χρήστης είναι δυνατόν να αλλάξει τα στοιχεία του μελλοντικά, κάτι που βέβαια δεν επηρεάζει άμεσα τα στοιχεία που έχουν ήδη συλλεγεί για το προφίλ του, εκτός κι αν ο ίδιος επιθυμεί δημιουργία από την αρχή του προφίλ που ήδη έχει. Οι πληροφορίες αποθηκεύονται στην κεντρικοποιημένη βάση δεδομένων και ανανεώνονται συνεχώς με το δυναμικό προφίλ του όπως θα δούμε στην επόμενη ενότητα. Όταν ο χρήστης βρίσκεται στη διαδικασία εγγραφής στο σύστημα του παρουσιάζονται όλες οι κατηγορίες του συστήματος και του ζητήται να δηλώσει την προτίμησή του για κάθε κατηγορία. Ο χρήστης καλείται να επιλέξει μία βαθμολογία για κάθε κατηγορία από -5 έως 5. Το -5 μεταφράζεται σαν η κατηγορία δε με αντιπροσωπεύει καθόλου ενώ το +5 σημαίνει πως η κατηγορία αντιπροσωπεύει απόλυτα το χρήστη. Η επιλογή του 0 σαν προτίμηση κατηγορίας μεταφράζεται σαν ουδέτερη στάση απέναντι στην κατηγορία. Εκμεταλλευόμενοι τις απαντήσεις των χρηστών μπορούμε να διαμορφώσουμε ένα αρχικό προφίλ για το χρήστη. Αυτό γίνεται ως εξής. Αρχικά δημιουργούμε εγγραφές για τις κατηγορίες που αρέσουν στο χρήστη και γι αυτές που ο χρήστης δεν προτιμά. Αυτό θα μας βοηθήσει να κάνουμε ένα πρώτο ξεκαθάρισμα των άρθρων ανάμεσα σε αυτά που ο χρήστης θέλει να δει και σε αυτά που δεν τον ενδιαφέρουν, ανάλογα με τις γενικές κατηγορίες που έχει επιλέξει. Ο χρήστης όμως δεν επιλέγει απλώς τι θέλει να βλέπει και τι δε θέλει. Έχει δώσει και κάποια βαθμολογία για κάθε κατηγορία. Χρησιμοποιώντας αυτά τα δεδομένα μπορούμε να δημιουργήσουμε μία πιο αναλυτική περιγραφή του προφίλ. Το αναλυτικό προφίλ όπως έχει ήδη αναφερθεί περιλαμβάνει λίστες με λέξεις κλειδιά όπως αυτές που υπάρχουν για τις κατηγορίες που δείχνουν ποιες λέξεις κλειδιά ενδιαφέρουν το χρήστη και ποιες δεν τον αφορούν. Σε αυτή την περίπτωση επιτρέπονται τόσο θετικά βάρη όσο και αρνητικά. Ο υπολογισμός των βαρών για τις λέξεις κλειδιά του χρήστη υπολογίζονται από τον αλγόριθμο 8.1.1.

Αλγόριθμος 8.1.1 extract_user_keywords

```
for all (selection s) do
  if (s!=0) then
    Keyword_name_usr = select 20*s keywords from category keywords;
    Keyword_weight_usr = select (2*s*relative frequency) from category keywords;
  else
    Keyword_name_usr = select 10 keywords from category keywords;
    Keyword_weight_usr = select relative_frequency from category.keywords;
  end if
  Insert_into_user_profile_keyword_name_usr, keyword_weight_usr;
  if exists then
    Update_user_profile_set_keyword_weight += keyword_weight_usr where keyword_name = ke-
    yword_name_usr;
  end if
end for
```

Υποθέτουμε ότι ο χρήστης κάνει κάποιες επιλογές για τις κατηγορίες και επιλέγει από -5 έως 5. Από αυτές τις επιλογές επιλέγουμε $20s$ λέξεις κλειδιά, όπου s είναι η επιλογή του χρήστη ($s \in [-5...5]$) από τη λίστα με τις λέξεις κλειδιά που αφορούν την κατηγορία, όπως ο πίνακας που είδαμε παραπάνω. Εν συνεχεία, επιλέγουμε τη σχετική συχνότητα κάθε λέξης και την πολλαπλασιάζουμε με $2s$. Αν για παράδειγμα ο χρήστης έχει επιλέξει για μία κατηγορία την επιλογή -3 και μία συγκεκριμένη λέξη κλειδί για την κατηγορία έχει σχετική συχνότητα 0,12 τότε στον πίνακα του χρήστη η συγκεκριμένη λέξη θα πάρει σχετική συχνότητα -0,12. αυτός ο αριθμός μας δείχνει και το πόσο ο χρήστης ενδιαφέρεται για τη συγκεκριμένη λέξη κλειδί. Στο παράδειγμα που δείξαμε ο χρήστης δεν ενδιαφέρεται για τη συγκεκριμένη λέξη. Πραγματοποιώντας αυτή τη διαδικασία καταλήγουμε σε μία αρχική λίστα με λέξεις κλειδιά και σχετικές συχνότητες για το χρήστη οι οποίες μας δίνουν τα παρακάτω στοιχεία:

- Πολλές λέξεις κλειδιά από τις κατηγορίες που έχει επιλέξει ο χρήστης με μεγάλο σκόρ, είτε θετικό είτε αρνητικό και παράλληλα πολύ λίγες λέξεις από τις κατηγορίες που έχει δηλώσει ο χρήστης με χαμηλό σκορ. Πρόκειται για κατηγορίες που είναι αδιάφορες στο χρήστη και άρα, λέξεις κλειδιά από αυτές τις κατηγορίε δεν είναι απαραίτητες για το προφίλ του χρήστη.

- Μεγάλη θετική τιμή για τις σχετικές συχνότητες των λέξεων κλειδιών που ανήκουν στις κατηγορίες

που έχει επιλέξει ο χρήστης με μεγάλο σκορ και μεγάλη απόλυτα αρνητική τιμή για τις σχετικές συχνότητες των λέξεων κλειδιών που ανήκουν σε κατηγορίες που έχει επιλέξει ο χρήστης με πολύ μικρό σκορ.

Αυτά τα στοιχεία μπορούν να μας δώσουν πληροφορίες για να εξάγουμε τα παρακάτω στοιχεία:

- Επιλογή κειμένων από τις κατηγορίες που ενδιαφέρουν το χρήστη
- Αποφυγή επιλογής κειμένων από κατηγορίες που δεν ενδιαφέρουν το χρήστη
- Επιλογή κειμένων από κατηγορίες που ενδιαφέρουν το χρήστη ενώ παράλληλα δεν ανήκουν σε κατηγορίες που δεν ενδιαφέρουν το χρήστη (να θυμίσουμε πως ένα κείμενο ανήκει σε πολλές κατηγορίες)
- Ξεκαθάρισμα των αποτελεσμάτων του μηχανισμού αυτόματης εξαγωγής περίληψης προσθέτοντας τον παράγοντα προσωποποίησης.

Η προαναφερθείσα διαδικασία, συμπεριλαμβανομένης και της κατασκευής της λίστας με τις λέξεις κλειδιά πραγματοποιήθηκε προκειμένου να έχουμε κάποια πρώτα στοιχεία για το αρχικό προφίλ του χρήστη. Στη συνέχεια θα περάσουμε στην κατασκευή του δυναμικού προφίλ χρήστη, το οποίο μεταβάλλεται με τη χρήση της υπηρεσίας RSS του μηχανισμού. Είναι σημαντικό το γεγονός πως όσο περισσότερο χρησιμοποιεί ο χρήστης την υπηρεσία RSS που του προσφέρει η μηχανισμός, τόσο καλύτερα διαμορφώνεται το προφίλ του.

Δυναμική διαμόρφωση προφίλ χρήστη

Όσο ο χρήστης χρησιμοποιεί την υπηρεσία, τόσο καλύτερα διαμορφώνεται το προφίλ του από τα στοιχεία που συλλέγονται από τις επιλογές του. Όπως έχουμε ήδη αναφέρει το στοιχείο που ελέγχεται για τη δυναμική διαμόρφωση του προφίλ του χρήστη είναι οι επιλογές του μόλις του εμφανίζονται άρθρα, δηλαδή αν και ποια άρθρα θα επιλέξει να δει ολοκληρωμένα στο δικτυακό τους τόπο.

Από το χρήστη του συστήματός μας περιμένουμε όταν του εμφανιστούν τα τελευταία 20 άρθρα, κάποια από αυτά να τα διαβάσει και άλλα να μην τα δει καθόλου. Και οι δύο αυτές αντιδράσεις κάτι μπορεί να σημαίνουν όμως και γι αυτό κάθε τέτοιο στοιχεία είναι αντικείμενο μελέτης για το μηχανισμό μας. Αυτό που μπορούμε να καταλάβουν δημιουργώντας εικονικά προφίλ στο μηχανισμό μας είναι πως ο χρήστης θα επιλέξει να διαβάσει τα άρθρα που τον ενδιαφέρουν ενώ στα υπόλοιπα δε θα δώσει σημασία. Αυτή τη συμπεριφορά χρήστη την καταγράφουμε και την εκμεταλλευόμαστε προκειμένου να διαμορφώσουμε το προφίλ του. Από τα άρθρα που παρουσιάζουμε στο χρήστη επιλέγουμε τις λέξεις κλειδιά. Για κάθε άρθρο που επιλέγει ο χρήστης να διαβάσει προσθέτουμε τις συγκεκριμένες λέξεις κλειδιά στο προφίλ του βάση της σχετικής συχνότητας που παρουσιάζουν στο συγκεκριμένο άρθρο. Πρόκειται για μία πολύ μεγάλη σχετική συχνότητα κάτι που είναι επιθυμητό καθότι πρόκειται για λέξεις κλειδιά σε ένα άρθρο που ενδιαφέρει το χρήστη. Για τα άρθρα που επιλέγει ο χρήστης παρατηρείται αλλαγή στο προφίλ του το οποίο τροποποιείται κατάλληλα ώστε να περιέχει τα keywords του κειμένου. Αν αυτά υπήρχαν, το βάρος τους στο προφίλ του συγκεκριμένου χρήστη αυξάνεται προσθέτοντας τη σχετική συχνότητα εμφάνισής τους σε αυτό.

8.2 Υλοποίηση σε συσκευές μικρού μεγέθους

Όπως έχει επισημανθεί και σε προηγούμενες ενότητες, οι περιορισμοί που θέτουν οι συσκευές μικρού μεγέθους στην προβολή και εύκολη διαχείριση του περιεχομένου, μας επιβάλλουν τον περιορισμό του μεγέθους της απάντησης σε συνάρτηση πάντα με τις δυνατότητες της συσκευής. Γνωρίζοντας τον χρήστη και αναγνωρίζοντας τη συσκευή που χρησιμοποιεί, η διαδικασία περίληψης, χρησιμοποιώντας έναν πίνακα της βάσης δεδομένων όπου κρατούνται τα συνήθη μεγέθη συσκευών και οι δυνατότητες απεικόνισης τους σε χαρακτήρες, αντιστοιχίζει την συσκευή με το επιθυμητό πλήθος χαρακτήρων που πρέπει να έχει η περίληψη ενός άρθρου για εύληπτη απεικόνιση στη συσκευή του χρήστη. Προφανώς όσο μικρότερη σε μέγεθος η συσκευή, τόσο πιο περιορισμένη θα είναι και η περίληψη του άρθρου. Η διαδικασία αυτή αξιοποιεί τη δυνατότητα του μηχανισμού εξαγωγής περίληψης για καθορισμό στα ορίσματα εισόδου του επιθυμητού μεγέθους της περίληψης σε σύνολο χαρακτήρων. Σε κάθε περίπτωση επιδιώκεται το μέγεθος της περίληψης να μην είναι λιγότερο από 3 προτάσεις ή 80 χαρακτήρες ώστε να υπάρχει ένας, έστω και μικρός, βαθμό κατανόησης του περιεχομένου άρθρου ακόμα και από χρήστες με πολύ μικρές συσκευές.

8.2.1 Αποστολή απάντησης στο χρήστη

Το πλήθος των άρθρων τα οποία περιλήπτονται πριν αποσταλούν σε μορφή RSS στον τελικό χρήστη εξαρτάται από τις προτιμήσεις του χρήστη. Επιλέγονται άρθρα που ανήκουν με μεγάλη συσχέτιση στις κατηγορίες που ενδιαφέρεται ο χρήστης και που ο χρόνος άφιξής τους στο σύστημα δεν ξεπερνά το τελευταίο 24-ωρο. Παράλληλα στέλνονται όχι πάνω από 10 άρθρα μέσω του RSS πίσω στον χρήστη και αυτό γιατί δεν επιθυμούμε ο χρήστης να 'υπερφορτωθεί' με άρθρα κάτι που ενδεχωμένος να κουράσει τον χρήστη. Έχοντας αυτούς τους περιορισμούς υπ' όψιν, ο μηχανισμός τροφοδοτεί τους χρήστες του συστήματος με περιλήψεις πρόσφατων άρθρων, που δεν ξεφεύγουν πολύ σε πλήθος αλλά και ανταποκρίνονται στις προτιμήσεις των χρηστών. Η απάντηση, όπως έχουμε αναφέρει, στέλνεται μέσω του προτύπου RSS 2.0 και είναι αναγνώσιμο από κάθε εφαρμογή χρήστη που υποστηρίζει αυτό το πρότυπο. Το τελευταίο ισχύει και για τις συσκευές μικρού μεγέθους οι δυνατότητες των οποίων αυξάνονται διαρκώς.

ΚΕΦΑΛΑΙΟ 9

Προδιαγραφές Και Χρήση Του Συστήματος

Reason is immortal, all else mortal.

Pythagoras, from Diogenes Laertius, Lives of Eminent Philosophers, Greek mathematician, philosopher, & scientist

Στο κεφάλαιο αυτό παρουσιάζονται οι προδιαγραφές του συστήματος που αναπτύχθηκε ώστε αυτό να είναι σε θέση να λειτουργεί σωστά και να παράχει αποτελέσματα που έχουν αξία. Επίσης δίνονται και ορισμένα στοιχεία που έχουν να κάνουν με τις απαιτήσεις του μηχανισμού σε υλικό και λογισμικό ώστε να μπορεί να λειτουργεί αποτελεσματικά.

9.1 Προδιαγραφές

9.1.1 Συλλογή άρθρων και εξαγωγή χρήσιμου κειμένου

Η λειτουργία του συστήματος ξεκινά με τον μηχανισμό ανάκτησης δεδομένων από το διαδίκτυο ο οποίος τρέχει ανεξάρτητα από τα υπόλοιπα υποσυστήματα που έχουν αλληλεπίδραση με τον χρήστη. Σε αυτόν περιλαμβάνονται η συλλογή άρθρων από τον ιστό και η εξαγωγή του χρήσιμου κειμένου από αυτά. Η λειτουργία είναι αυτοματοποιημένη ώστε να αλληλεπιδρά με τη ΒΔ και η ανθρώπινη επίδραση μπορεί να είναι μόνο έμμεση. Κατ' αρχάς καθορίζονται στη ΒΔ τα urls των RSS feeds των news portals τα οποία πρέπει να διαπεράσει ο crawler. Είναι εύλογο πως υπόκειται στον διαχειριστή του συστήματος ο καθορισμός έγκυρων urls για την τροφοδότηση του μηχανισμού με άρθρα. Ο μηχανισμός εξαγωγής χρήσιμου κειμένου είναι σχεδιασμένος ώστε να εξάγει κείμενα άρθρων από τη σελίδα· δεν έχει επομένως νόημα, και για την ακρίβεια γεμίζει τη ΒΔ με 'σκουπίδια', η εισαγωγή urls από RSS feeds που δεν περιέχουν. Παρόμοια, πρέπει να αποφεύγεται η χρήση urls που δεν υπάρχουν dead links καθώς οδηγούν τον Java crawler και όλο συνολικά το σύστημα σε χάσιμο χρόνου.

9.1.2 Προεπεξεργασία κειμένου

Δεδομένου ότι ο μηχανισμός προεπεξεργασίας κειμένου είναι αυτοματοποιημένος ώστε να αλληλεπιδρά με τα κείμενα της ΒΔ, η ορθή λειτουργία του εναπόκειται στην ορθή κατάσταση της ΒΔ και τις συναλλαγές που γίνονται με αυτή. Τα δεδομένα που διαβάζονται από τη ΒΔ μορφοποιούνται σε XML μορφή και δίνονται στον μηχανισμό. Δεδομένου ότι όλα τα απαραίτητα πεδία των πινάκων της ΒΔ περιέχουν σωστές πληροφορίες, η εξαγωγή κωδικολέξεων προχωράει βάσει αυτών. Πρέπει να σημειωθεί επίσης ότι η διαδικασία της προεπεξεργασίας κειμένου (αφαίρεση στίξης και αριθμών, stopwords, stemming) εκτελείται εμμέσως

με την κλήση των υποσυστημάτων κατηγοριοποίησης και περίληψης κειμένου αν αυτό είναι απαραίτητο. Σε κάθε περίπτωση πάντως, όταν ένα νέο κείμενο έχει προστεθεί στη ΒΔ μπορεί να εισαχθεί στον μηχανισμό προεπεξεργασίας. Αν αυτό δεν έχει γίνει (ασύγχρονα) από το σύστημα μέχρι να ζητηθεί περίληψη ή κατηγοριοποίησή του, γίνεται προεπεξεργασία του εκείνη τη στιγμή. Τα αποτελέσματα του υποσυστήματος εξαγωγής κωδικολέξεων, όπως έχουμε πει, είναι σε μορφή XML για λόγους διασυνδεσιμότητας. Αυτά αποθηκεύονται στους κατάλληλους πίνακες της ΒΔ του συστήματος για να είναι διαθέσιμα στα υποσυστήματα που ακολουθούν.

9.1.3 Κατηγοριοποίηση και εξαγωγή περίληψης

Τα υποσυστήματα κατηγοριοποίησης και εξαγωγής περίληψης είναι σχεδιασμένα ώστε να δέχονται είσοδο XML τα δεδομένα της οποίας μπορούν να τα διαβάσουν είτε από τη ΒΔ, είτε από κατάλληλα μορφοποιημένο XML αρχείο. Όπως έχει ήδη αναφερθεί, η διαδικασία που ακολουθείται μετά την προεπεξεργασία κειμένου είναι: προσπάθεια για κατηγοριοποίηση του κειμένου βάσει κάποιων κριτηρίων και της βάσης γνώσης που έχουμε, αν η κατηγοριοποίηση είναι επιτυχής (το κείμενο είναι πολύ σχετικό με μία κατηγορία), προχωρούμε σε εξαγωγή γενικής περίληψης υποβοηθούμενη από την κατηγορία του κειμένου. Αν η κατηγοριοποίηση δεν είναι εφικτή, προχωρούμε σε εξαγωγή γενικής περίληψης και επιχειρούμε την κατηγοριοποίηση αυτής. Αν η δεύτερη απόπειρα κατηγοριοποίσης δώσει καλύτερα αποτελέσματα, αποθηκεύουμε αυτά στη ΒΔ, αλλιώς τα πρώτα. Φυσικά τα υποσυστήματα μπορούν να κληθούν και αυτόνομα, π. χ. να ζητήσουμε περίληψη ή κατηγοριοποίηση ενός άρθρου που έχουμε στην κατοχή μας.

Για τη διαδικασία κατηγοριοποίησης βασιζόμαστε στον αλγόριθμο κατηγοριοποίησης SVM και πιο συγκεκριμένα στο σχήμα LSI. Στο σύστημα εφαρμόζεται μία πρωτότυπη μέθοδος κατηγοριοποίησης που βασίζεται στην ανάλυση προτάσεων και όχι ολόκληρων των παραγράφων των κειμένων. Πιο συγκεκριμένα, η ανάλυση είναι διαφορετική για τους διαφορετικούς χρήστες. Όσο μεγαλύτερη είναι η αφαίρεση πληροφορίας σε τόσο λιγότερες προτάσεις ενός κειμένου πραγματοποιείται κατηγοριοποίηση του κειμένου και συνεπώς η κατηγορία στην οποία εντάσσεται ένα κείμενο είναι πιο γενική. Η παραπάνω διαδικασία έχει σαν αποτέλεσμα να δημιουργηθεί πολλαπλού είδους κατηγοριοποίηση στα κείμενα τα οποία θα διαθέτει το σύστημα με αποτέλεσμα να είναι διαφορετικά τα αποτελέσματα για κάθε χρήστη ανάλογα με τη λεπτομέρεια της αναζήτησης που πραγματοποιούν. Το ένα είδος κατηγοριοποίησης θα είναι καθαρά αλγοριθμικό ενώ το δεύτερο κομμάτι θα βασίζεται κυρίως στις προσωπικές επιλογές του χρήστη, οι οποίες δημιουργούν κατηγορίες αφαίρεσης πληροφορίας.

Έχει ήδη αναφερθεί αρκετές φορές ποια είναι η λειτουργία του μηχανισμού κατηγοριοποίησης. Αξίζει όμως να τονίσουμε κάποια βασικά στοιχεία της λειτουργίας αυτού του μηχανισμού. Ο μηχανισμός αυτός από τη στιγμή που θα αρχικοποιηθεί με ένα σύνολο πρότυπων κειμένων για τη δημιουργία μίας κατηγορίας μπορεί να λειτουργεί ανεξάρτητα από το υπόλοιπο σύστημα κατηγοριοποιώντας συνεχώς κείμενα. Είναι πολύ βασικό για την καλή λειτουργία του συστήματος να υπάρχουν συνεχώς κείμενα προς κατηγοριοποίηση προκειμένου να μη μένει ο μηχανισμός αδρανής αλλά και να ανανεώνεται η βάση γνώσης με επικαιροποιημένα κείμενα.

Θα πρέπει σε αυτό το σημείο να τονίσουμε ότι από άποψη κώδικα έγινε προσπάθεια να καλυφθούν οι συνηθέστερες αιτίες εξαιρέσεων (exceptions), όπως είναι κατανοητό όμως η έκταση του μηχανισμού δεν επιτρέπει τον προσδιορισμό κάθε 'επικίνδυνης' κατάστασης που μπορεί να επιφέρει δυσάρεστα αποτελέσματα.

9.2 Απαιτήσεις του συστήματος

Λογισμικό και βιβλιοθήκες

Για την ανάπτυξη του συστήματος χρησιμοποιήθηκαν τα παρακάτω πακέτα λογισμικού και βιβλιοθήκες 9.1:

Η ανάπτυξη του συστήματος έγινε εξ' ολοκλήρου σε open source λογισμικό και λειτουργικό σύστημα Gentoo Linux.

Kdevelop-3.4.1 [19]
NetBeans-5.5.1 [24]
GCC-4.1.2 [11]
MySQL-5.0.44 [23]
Apache-2.0.58 [2]
PHP-5.2.4 [28]
Java-1.5.0 [18]
Boost-filesystem-1.33.1 [4]
Boost-regex-1.33.1 [4]
cgicc-3.2.3 [6]
mysql++-2.2.2-r1 [22]

Πίνακας 9.1: Σύνθεση υλικού για ανάπτυξη του συστήματος

Υλικό

Το σύστημα που αναπτύχθηκε δεν έχει υψηλές απαιτήσεις υλικού. Μπορεί να στηθεί σε κάποιον υπολογιστή γενιάς Pentium II και νεότερο. Φυσικά εάν οι απαιτήσεις μας έχουν να κάνουν με ένα σύστημα που θα πραγματοποιεί real time κατηγοριοποίηση και εξαγωγή προσωποποιημένης περίληψης κειμένων είναι εύλογο να χρησιμοποιηθεί ένα πιο σύγχρονο σύστημα στο οποίο η ΒΔ (η οποία και αποτελεί το bottleneck του συστήματος λόγω των πολλών συναλλαγών) θα έχει καλύτερους χρόνους εξυπηρέτησης. Για την ανάπτυξη των μηχανισμών χρησιμοποιήθηκε η παρακάτω σύνθεση υλικού (Πίνακας 9.2):

CPU	Intel Centrino 1.6 GHz
RAM	1256MB 333MHz
Cache	2048KB
Hard Disk	80GB, 7200rpm

Πίνακας 9.2: Σύνθεση υλικού για ανάπτυξη του συστήματος

ενώ για την καθημερινή λειτουργία του συστήματος χρησιμοποιείται η παρακάτω σύνθεση (Πίνακας 9.3):

CPU	Intel Pentium 4 2.4 GHz
RAM	256MB 333MHz
Cache	512KB
Hard Disk	80GB, 720rpm

Πίνακας 9.3: Σύνθεση υλικού για καθημερινή λειτουργία του συστήματος

ΚΕΦΑΛΑΙΟ 10

Πειραματικά αποτελέσματα και αξιολόγηση

> I think computer viruses should count as life. I think it says something about human nature that the only form of life we have created so far is purely destructive. We've created life in our own image.
>
> *Stephen Hawking, English cosmologist and physicist*

Η ανάπτυξη του συστήματος που έγινε στα πλαίσια της παρούσας εργασίας έγινε τμηματικά με κάθε module αυτού να αναπτύσσεται ξεχωριστά από τα υπόλοιπα. Την ανάπτυξη του καθενός τμήματος ακολουθούσε και μια διαδικασία αξιολόγησης του ώστε: α) να εντοπισθεί η αποτελεσματικότητά του ως ξεχωριστή οντότητα και β)να προσδιοριστούν οι απαραίτητες παράμετροι που πρέπει να χρησιμοποιηθούν σε κάθε βήμα ώστε ο μηχανισμός, ως σύνολο, να παράγει το βέλτιστο αποτέλεσμα. Ακολουθεί μια αναλυτική παρουσίαση των πειραματικών διαδικασιών και αξιολογήσεων που έλαβαν μέρος και που αφορούν στα βασικά υποσυστήματα του μηχανισμού: τη διαδικασία του keyword extraction, τους μηχανισμούς κατηγοριοποίησης και περίληψης μαζί με τις μεταξύ τους αλληλεπιδράσεις, και τέλος, το υποσύστημα παρουσίασης πληροφορίας στο χρήστη συσκευής μικρού μεγέθους.

10.1 Μηχανισμός εξαγωγής κωδικολέξεων

Σε αρχικό στάδιο υλοποίησης του μηχανισμού, εξετάσθηκε η αποτελεσματικότητα της διαδικασίας εξαγωγής keywords από διάφορες μορφές κειμένου. Με αυτό τον τρόπο, προσπαθήσαμε να αξιολογήσουμε τη διαδικασία αλλά και να θέσουμε κάποιες αρχικές παραμέτρους οι οποίες θα χρειαστούν για την λειτουργία του μηχανισμού ως σύνολο.

Δεδομένου ότι ο μηχανισμός εξαγωγής keywords είναι ένα ανεξάρτητο υποσύστημα, ο τύπος των κειμένων εισόδου μπορεί να διαφέρει κατά πολύ. Έτσι χρησιμοποιήθηκαν e-mails, άρθρα νέων αλλά και ερευνητικές εργασίες papers ως είσοδος. Για κάθε μία από αυτού του είδους την είσοδο, διεξάγαμε πειραματική διαδικασία ώστε να εντοπιστεί ποιο είναι το ελάχιστο δυνατό μήκος από keywords του αρχικού κειμένου που πρέπει να κρατηθούν, ώστε το αποτέλεσμα που προκύπτει να μη χάνει σημαντικά το νόημα του κειμένου. Για την διαδικασία αυτή, αξιολογήθηκαν δύο παράγοντες:

- ποιο είναι το ελάχιστο μήκος λέξεων που πρέπει να κρατηθεί
- τι ποσοστό των τελικών keywords πρέπει να κρατηθεί.

Για να 'μετρηθεί' η διαφορά του νοήματος μεταξύ δύο κειμένων (δηλ. εκείνου στο οποίο έχουμε ελάχιστο μήκος λέξεων 4 και εκείνου που έχουμε ελάχιστο μήκος λέξεων 6), χρησιμοποιήθηκε μια απλή έκδοση του SVM αλγορίθμου [127].

Αν υποθέσουμε ότι έχουμε έναν πίνακα a με όλα τα keywords και τις συχνότητές τους για το κείμενο A, και έναν πίνακα b του κειμένου B, τότε μπορούμε να υπολογίσουμε τη συσχέτιση μεταξύ των δύο κειμένων ως:

$$x = a * b \tag{10.1.1}$$

$$y = |a| * |b| \tag{10.1.2}$$

$$z = x/y \tag{10.1.3}$$

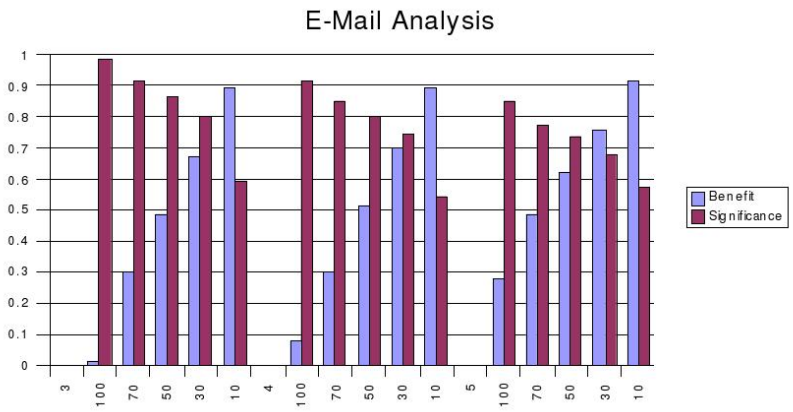
$$r = sin(z) \tag{10.1.4}$$

όπου x είναι το εσωτερικό γινόμενο των πινάκων a και b και y το γινόμενο των νορμών (2) του A και του B. Όπως μπορούμε να δούμε από τις προηγούμενες εξισώσεις, το r κινείται μεταξύ των τιμών μηδέν και ένα. Όταν το r είναι μηδέν, τότε οι πίνακες a και b είναι εντελώς ασυσχέτιστοι μεταξύ τους, ενώ όταν το r είναι ένα, οι πίνακες είναι εντελώς όμοιοι. Αυτό σημαίνει ότι όταν το r είναι κοντά στο ένα, τότε έχουμε υψηλή συσχέτιση μεταξύ των κειμένων που αναπαρίστανται μέσω των πινάκων.

Με σκοπό να περιοριστεί ακόμη περισσότερο ο αριθμός των keywords του κειμένου, κρατήσαμε μόνο ένα ποσοστό αυτών και επανυπολογίσαμε από τη σχέση 10.1.4 την συσχέτιση μεταξύ των keywords του αρχικού κειμένου και του ποσοστού των keywords που κρατήθηκε.

10.1.1 Πειραματισμός με $e-mails$

Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα που προέκυψαν από την πειραματική διαδικασία με κείμενα ηλεκτρονικού ταχυδρομείου. Κατά τη διάρκεια της πειραματικής διαδικασίας χρησιμοποιήθηκε ελάχιστο μήκος λέξεων τριών, τεσσάρων και πέντε γραμμάτων. Τα αποτελέσματα συνοψίζονται στην γραφική απεικόνιση του Σχήματος 10.1



Σχήμα 10.1: Ανάλυση κειμένων ηλεκτρονικού ταχυδρομείου.

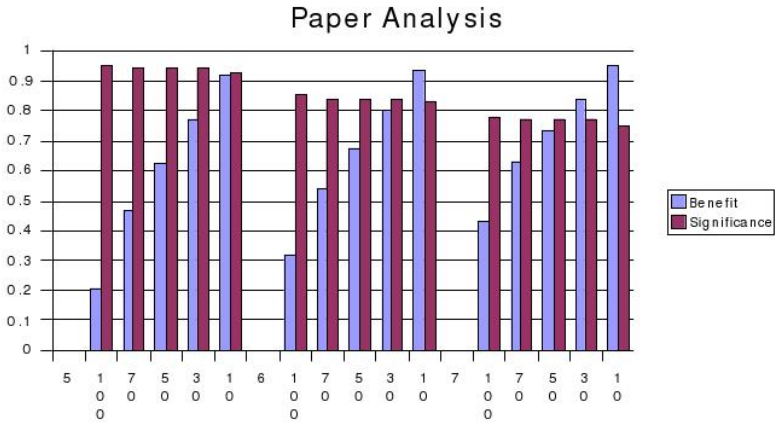
Όπως φαίνεται και στο Σχήμα 10.1, έχουμε περιορίσει το ελάχιστο μήκος των λέξεων σε 3, 4, 5 και περισσότερους χαρακτήρες και κρατήσει ένα ποσοστό των keywords που απομένουν. Μειώνοντας το ελάχιστο μήκος λέξεων σε 3 γράμματα και κρατώντας το 70% των εξαγόμενων keywords, έχουμε ένα όφελος περίπου 30% των keywords του αρχικού κειμένου και η ομοιότητα των δύο κειμένων είναι πάνω από 90%.

Αυτό που μας ενδιαφέρει είναι η συσχέτιση μεταξύ του αρχικού κειμένου και των εξαγομένων keywords. Έτσι αποφασίσαμε να κρατήσουμε το επίπεδο της συσχέτισης στο 85% αφού είναι προφανές ότι τα keywords

που απομένουν είναι αντιπροσωπευτικά του αρχικού κειμένου. Ο περιορισμός αυτός σημαίνει ότι το ελάχιστο μήκος λέξεων και το ποσοστό των keywords που προκύπτουν από το προηγούμενο διάγραμμα, μπορεί να είναι: 3/100%, 3/70%, 3/50%, 4/100%, 4/70% και 5/100% αντίστοιχα. Το όφελος από τα ζευγάρια αυτά είναι 1%, 29%, 48%, 8%, 30% και 28% αντίστοιχα. Ο λόγος όφελος / ομοιότητα είναι 0.01, 0.33, 0.56, 0.09, 0.35 και 0.33 για καθένα από τα ζεύγη που αναφέρθηκαν. Αυτό σημαίνει ότι το καλύτερο ζεύγος μοιάζει να είναι το 3/50% για την ανάλυση κειμένων ηλεκτρονικού ταχυδρομείου, μειώνουμε δηλαδή το ελάχιστο μήκος λέξεων σε 3 γράμματα και κρατάμε τις μισές από τις κωδικολέξεις που προκύπτουν από την ανάλυση. Πρέπει να αναφερθεί επίσης ότι τα keywords βρίσκονται σε φθίνουσα σειρά διάταξης σε σχέση με τη συχνότητα εμφάνισης, πριν κρατηθεί το κατάλληλο ποσοστό.

10.1.2 Πειραματισμός με *papers*

Σε αυτή την ενότητα παρουσιάζουμε τα αποτελέσματα του μηχανισμού προεπεξεργασίας όταν επεξεργάζεται papers. Στην ανάλυση χρησιμοποιήθηκε ελάχιστο μήκος λέξεων 5, 6, 7 και περισσοτέρων γραμμάτων. Στο Σχήμα 10.2 παρουσιάζονται τα αποτελέσματα που προέκυψαν μέσω της πειραματικής διαδικασίας.



Σχήμα 10.2: Ανάλυση κειμένων δημοσιεύσεων.

Όπως μπορούμε να δούμε από τη γραφική παράσταση του Σχήματος 10.2, κρατήθηκε ελάχιστο μήκος λέξεων 5, 6, 7 και περισσότεροι χαρακτήρες και στη συνέχεια κρατήθηκε ένα ποσοστό των keywords για καθένα από τον περιορισμό μήκους λέξεων. Όπως μπορούμε να δούμε, τα αποτελέσματα δεν επηρεάζονται (σημαντικά) από τον παράγοντα ποσοστού κράτησης των λέξεων. Αυτό μπορεί να εξηγηθεί ως εξής: τα κείμενα που επεξεργάζεται ο μηχανισμός εξαγωγής keywords σε αυτή την περίπτωση, περιέχουν περισσότερες από 900 μοναδικές λέξεις οι οποίες εμφανίζονται πολλές φορές μέσα στο κείμενο και αυτό γιατί τα papers έχουν ένα συγκεκριμένο θεματικό πεδίο, με αποτέλεσμα, η επαναληπτικότητα των όρων είναι αναπόφευκτη. Το όριο της συσχέτισης ώστε να θεωρηθεί ότι το κείμενο δεν έχει χάσει το νόημά του, επιλέχθηκε να είναι το 80%. Αυτό σημαίνει ότι ο περιορισμός μήκους λέξεων για 7 ή περισσότερους χαρακτήρες μοιάζει να μην επιτυγχάνει το στόχο. Αντίθετα, με ελάχιστο μήκος λέξεων 5 ή 6 χαρακτήρων, το κείμενο που προκύπτει ξεπερνά σε συσχέτιση με το αρχικό κείμενο το όριο του 80% για την ομοιότητα.

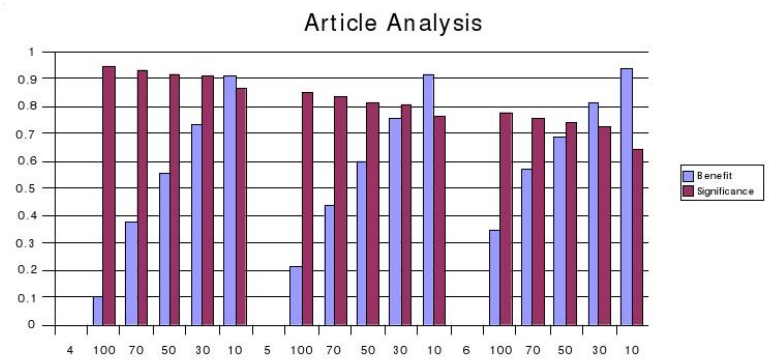
Το ζεύγος που αξιολογήθηκε ως βέλτιστο για να κρατηθεί, είναι το 6/10%, δηλαδή 6 χαρακτήρες ως ελάχιστο μήκος λέξεων και 10% των εξαγόμενων keywords, το οποίο μας οδηγεί σε 83% ομοιότητα και πάνω από 90% όφελος.

10.1.3 Πειραματισμός με άρθρα

Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα που προέκυψαν από την ανάλυση άρθρων ειδήσεων του διαδικτύου. Σε αυτή την περίπτωση κρατάμε ελάχιστο μήκος 4, 5, 6 και περισσοτέρων χαρακτήρων, και κρατάμε ένα ποσοστό των εξαγόμενων keywords για να βρούμε το καλύτερο ζεύγος ελάχιστου μήκους λέξης / ποσοστού των keywords το οποίο έχει καλά αποτελέσματα για την ομοιότητα και το όφελος που προκύπτει.

Το όριο για την ομοιότητα που τέθηκε είναι το 85%, κάτι που προέκυψε ύστερα από πειραματική διαδικασία με χρήση πολλών άρθρων και λειτουργία όλου του μηχανισμού (όχι μόνο του υποσυστήματος εξαγωγής κωδικολέξεων αλλά και των υποσυστημάτων περίληψης / κατηγοριοποίησης κειμένων). Ανεβάζοντας αυτό το ποσοστό στο 90%, οδηγούμαστε σε πάρα πολλά keywords κάτι που υπερφορτώνει τη βάση δεδομένων αλλά και τους μηχανισμούς εξαγωγής πληροφορίας που ακολουθούν.

Τα ζεύγη που μπορούν να περάσουν το όριο του 85%, μπορούν να βρεθούν μόνο στις περιπτώσεις που κρατούνται 4 και 5 χαρακτήρες ως ελάχιστο μήκος λέξεων. Πιο συγκεκριμένα, όλα τα ζεύγη που προκύπτουν από την χρήση 4 χαρακτήρων και η πρώτη επιλογή από τη χρήση 5 χαρακτήρων ικανοποιούν το όριο που αναφέρθηκε. Η πρώτη επιλογή από τη χρήση 5 χαρακτήρων, έχει πολύ μικρό όφελος (21%). Αντίθετα το ζεύγος 4/10% μας δίνει ομοιότητα πάνω από 85% και όφελος που ξεπερνάει το 90%. Αυτό σημαίνει ότι κόβουμε το 90% των μοναδικών keywords και αποθηκεύουμε μόνο ένα 10% αυτών που μας δίνουν πάνω από 85% ομοιότητα του τελικού κειμένου σε σχέση με το αρχικό. Τα παραπάνω συνοψίζονται και στο διάγραμμα του Σχήματος 10.3



Σχήμα 10.3: Ανάλυση άρθρων ειδήσεων από το διαδίκτυο.

10.1.4 Γενικά αποτελέσματα

Ύστερα από τον πειραματισμό με διάφορα είδη κειμένων, μπορούμε να αντιληφθούμε ότι τα διάφορα είδη κειμένων χρειάζονται διαφορετική αντιμετώπιση από τον μηχανισμό προεπεξεργασίας. Η απλή δομή και περιεκτικότητα των μηνυμάτων ηλεκτρονικού ταχυδρομείου είναι πολύ διαφορετική από την πολύπλοκη δομή των papers. Κάπου ενδιάμεσα βρίσκονται τα άρθρα ειδήσεων από το διαδίκτυο που μας απασχολούν και στην συγκεκριμένη εργασία.

Όπως μπορούμε να δούμε από τα αποτελέσματα που προέκυψαν, στα e-mails πρέπει να κρατηθούν όλα τα keywords με μικρό μάλιστα ελάχιστο μήκος λέξεων. Αντίθετα, στις δημοσιεύσεις, όπου οι λέξεις που χρησιμοποιούνται είναι συνήθως επίσημες και μεγάλες σε μήκος, μπορούμε να ωφεληθούμε από αυτό και να θέσουμε υψηλότερα το ελάχιστο μήκος λέξεων και να κρατήσουμε ένα σχετικά μικρό ποσοστό των keywords που προκύπτουν για να αναπαραστήσουμε το κείμενο.

Αναμέναμε ότι κερδίζοντας σε σημαντικότητα τις τελικής λίστας keywords θα οδηγούμασταν σε μείωση του οφέλους. Αντίθετα, από τα αποτελέσματα προέκυψε ότι μπορούμε να κρατήσουμε ένα υψηλό ποσοστό και για δύο αυτές παραμέτρους. Αυτό σημαίνει ότι καταφέραμε, για τα διάφορα είδη κειμένων, να καταλήξουμε σε ένα τελικό μέγεθος λίστας keywords το οποίο ήταν 80% περίπου μικρότερο από την αρχική λίστα των keywords και συσχετίζονταν με αυτή σε ποσοστό πάνω από 80%. Με άλλα λόγια, για ένα κείμενο 5000 λέξεων, κρατώντας μόνο 20% αυτών (100 λέξεις) έχουμε μια καλή αναπαράσταση του αρχικού κειμένου η οποία μπορεί να αποθηκευθεί στη βάση δεδομένων για να αξιοποιηθεί από τους μηχανισμούς ανάκτησης πληροφορίας που ακολουθούν (περίληψη, κατηγοριοποίηση). Επομένως, δεν είναι αναγκαία η δεικτοδότηση ολόκληρου του αρχικού κειμένου και άρα, με τη χρήση ενός μικρού μόνο μέρους του, μειώνουμε α) τις απαιτήσεις για αποθήκευση δεδομένων και β) την πολυπλοκότητα και τους χρόνους εκτέλεσης των μηχανισμών που ακολουθούν.

10.2 Μηχανισμοί κατηγοριοποίησης και περίληψης

Κάθε μια από τις εξισώσεις (5.2.1),(5.2.6) και (5.2.8) για την βαθμολόγηση των προτάσεων ελέγχθηκε σε κάποια προκατηγοριοποιημένα (από ανθρώπους) κείμενα. Τα αποτελέσματα του μηχανισμού δείχνουν να είναι επαρκή σε σύγκριση με ήδη υπάρχοντα συστήματα. Ο βασικός μας στόχος είναι να παρουσιάζουμε μια προσωποποιημένη περίληψη άρθρων στον τελικό χρήστη και επομένως οι περιλήψεις που προκύπτουν βάσει των σχέσεων (5.2.1) και (5.2.6) δεν θα πρέπει να παράγουν περιλήψεις που διαφέρουν πολύ από ήδη υπάρχοντες αλγορίθμους. Η διαδικασία προσωποποίησης στην περίληψη δεν μπορεί να αξιολογηθεί σε σχέση με μια πρωτότυπη, ανθρώπινα παραγόμενη περίληψη αφού κάθε τέτοια εμπεριέχει τον υποκειμενικό ανθρώπινο παράγοντα. Ο μόνος πραγματικός εκτιμητής του συστήματος είναι ο τελικός χρήστης ο οποίος διαβάζει τις περιλήψεις.

Για την αξιολόγηση του αλγόριθμου περίληψης, εκτελέσθηκε πειραματική διαδικασία για την σύγκρισή του με τον MEAD αλγόριθμο περίληψης ο οποίος χρησιμοποιείται από την εφαρμογή του Microsoft Word. Οι προσωποποιημένες περιλήψεις που προέκυψαν από το σύστημα αξιολογήθηκαν από πέντε διαφορετικούς χρήστες οι οποίοι επιθυμούσαν να λάβουν μέρος στη δοκιμή.

10.2.1 Αξιολόγηση του μηχανισμού αυτόματης εξαγωγής περίληψης

Για να εξασφαλίσουμε ότι η διαδικασία πριν την εφαρμογή του παράγοντα προσωποποίησης παράγει επαρκή αποτελέσματα για τις περιλήψεις, αξιολογήσαμε τον μηχανισμό σε σχέση με τα αποτελέσματα από τον περιλήπτη του Microsoft Word. Τα αποτελέσματα συγκρίνονται με εξαγωγές του MEAD περιλήπτη σε 30 άρθρα συγκεντρωμένα από βασικά portals των Η.Π.Α και της Βρετανίας. Οι μετρικές που χρησιμοποιήθηκαν για τον υπολογισμό των αποτελεσμάτων είναι η ακρίβεια και η ανάκληση.

Πίνακας 10.1: Σύγκριση του αλγορίθμου περίληψης του συστήματος με τον περιλήπτη του *MS Word*

	MS Word		**Κατασκευασμένος Μηχανισμός**	
	Ακρίβεια	**Ανάκληση**	**Ακρίβεια**	**Ανάκληση**
Άρθρο 1	0,33	0,12	0,66	0,75
Άρθρο 2	0,12	0,25	0,75	0,66
Άρθρο 3	0,25	0,12	0,5	0,66
Άρθρο 4	0,25	0,12	0,75	0,5
Άρθρο 5	0,33	0,5	0,66	1
Άρθρο 6	0,33	0,25	0,66	0,75
Άρθρο 7	0,25	0,33	0,75	0,66

Από τα αποτελέσματα (Πίνακας 10.1) συνεπάγεται ότι ο μηχανισμός περίληψης που υλοποιήθηκε παράγει επαρκή αποτελέσματα συγκρινόμενος με δοκιμές που έγιναν με τον MEAD περιλήπτη, και σαφώς καλύτερα αποτελέσματα από τον περιλήπτη του MS Word. Προσθέτοντας τον παράγοντα κατηγοριοποίησης στη διαδικασία περίληψης, καταφέρνουμε να λάβουμε λίγο καλύτερα αποτελέσματα. Παρατηρούμε ότι η συνολική αύξηση είναι περίπου 10% σε σχέση με τα προηγούμενα αποτελέσματα όσον αφορά τις μετρικές της ακρίβειας και ανάκλησης. Η διαφορά οφείλεται στην διαδικασία κατηγοριοποίησης και, πιο συγκεκριμένα, στην προσθήκη της παραμέτρου k_3 στην εξίσωση εξαγωγής περίληψης. Η παράμετρος αυτή, επιτρέπει την υψηλότερη βαθμολόγηση των προτάσεων που περιέχουν keywords αντιπροσωπευτικά της κατηγορίας στην οποία ανήκει το άρθρο. Εάν ένα άρθρο δεν περιέχει πολλά keywords από την κατηγορία στην οποία ανήκει, δεν συμβαίνουν αλλαγές. Σε αυτή την περίπτωση, είναι αξιοσημείωτο να σημειωθεί ότι ύστερα από λίγο χρόνο (και ενώ νέα keywords προστίθενται στο σύστημα), όταν κάποιος προσπαθεί να έχει πρόσβαση στην περίληψη του συγκεκριμένου άρθρου, αυτή ανανεώνεται και οι μετρικές της ακρίβειας και ανάκλησης μετρώνται υψηλότερα σε σχέση με την πρώτη φορά της εξαγωγής περίληψης. Στον Πίνακα 10.2 οι μετρικές της ακρίβειας και ανάκλησης παρουσιάζονται για ένα συγκεκριμένο άρθρο και πως μεταβάλλονται όταν νέα άρθρα κατηγοριοποιούνται και πιο αντιπροσωπευτικά keywords για την κατηγορία προστίθενται στο σύστημα. Τα άρθρα 'καταφτάνουν' στο σύστημα κάθε μία ώρα αφού τα σημαντικά news portal ανανεώνουν το περιεχόμενό τους πολύ συχνά.

Από τα προηγούμενα στατιστικά στοιχεία, φαίνεται ότι ο μηχανισμός δεν είναι στατικός. Αντίθετα το σύστημα μπορεί να προσαρμόζεται δυναμικά και να ανανεώνει τις περιλήψεις που εξάγονται. Παράλληλα,

είναι αναμενόμενο το γεγονός ότι μετά την δημοσίευση ενός άρθρου κάποιου σημαντικού νέου, πολλά ακόμη άρθρα σχετικά με αυτό θα ακολουθήσουν. Αυτό σημαίνει ότι στα επόμενα 103 άρθρα μιας κατηγορίας που συλλέγονται από τον μηχανισμό στις επόμενες 78 ώρες, τουλάχιστον ένα θα είναι παρόμοιο με το πρώτο άρθρο είτε ως επανέκδοσή του είτε ως συμπλήρωμά του.

10.2.2 Αξιολόγηση του μηχανισμού εξαγωγής προσωποποιημένης περίληψης

Η αξιολόγηση μιας δυναμικά εξαγόμενης προσωποποιημένης περίληψης κειμένου δεν είναι μια διαδικασία που μπορεί να γίνει με χρήση μέτρων σύγκρισης. Το μέτρο που χρησιμοποιείται για να αξιολογηθούν οι εξαγόμενες περιλήψεις είναι η συσχέτιση μεταξύ της περίληψης και του άρθρου που παρατηρείται από τους χρήστες του μηχανισμού. Η διαδικασία που ακολουθήθηκε για να αξιολογηθούν τα αποτελέσματα της πειραματικής διαδικασίας ήταν: (α) δώσε στους χρήστες το πλήρες κείμενο του άρθρου, (β) δώσε στους χρήστες τις περιλήψεις που προέκυψαν τόσο από την εξίσωση (5.2.6), όσο και από την εξίσωση (5.2.8), και (γ) άφησε τους χρήστες να επιλέξουν ποια περίληψη θεωρούν ως περισσότερο αντιπροσωπευτική για το άρθρο που διάβασαν. Η αντίστροφη διαδικασία εξετάστηκε επίσης, δόθηκαν δηλαδή πρώτα οι περιλήψεις στους χρήστες, στη συνέχεια το κείμενο και τέλος οι χρήστες αποφάνθηκαν για το ποια περίληψη θεωρούν ως περισσότερο αντιπροσωπευτική για το πλήρες άρθρο που διάβασαν. Και στις δύο περιπτώσεις που αναφέρθηκαν οι απαντήσεις ήταν οι ίδιες.

Οι χρήστες που έλαβαν μέρος στην πειραματική διαδικασία μπορούν να χωριστούν σε τρεις ομάδες: (α) νέοι χρήστες του συστήματος, (β) παλιοί χρήστες του συστήματος αλλά με μικρή δραστηριότητα (το οποίο σημαίνει λίγα δεδομένα για προσωποποίηση), και (γ) προχωρημένοι χρήστες του συστήματος με υψηλή καθημερινή δραστηριότητα (το οποίο σημαίνει πολλά δεδομένα για προσωποποίηση). Σύμφωνα με αυτές τις κατηγορίες, τρεις διαφορετικές καταστάσεις παρατηρήθηκαν. Οι νέοι χρήστες του συστήματος εξέφρασαν την άποψη ότι οι περιλήψεις που τους δόθηκαν ήταν όμοιες, κάτι που είναι μια λογική παρατήρηση εφόσον το σύστημα δεν έχει αρκετή πληροφορία για την διαδικασία προσωποποίησης και επομένως, η βαθμολόγηση των προτάσεων για την περίληψη δεν επηρεάζεται από τον παράγοντα k_4 (που χρησιμοποιείται για την προσωποποίηση της περίληψης). Οι χρήστες της δεύτερης ομάδας επέλεξαν, με ποσοστό μεγαλύτερο του 80% των άρθρων, την περίληψη που εξήχθη από την εξίσωση (5.2.6) (χωρίς τον παράγοντα προσωποποίησης). Αυτό ήταν επίσης αναμενόμενο αφού το προφίλ των χρηστών αυτών (με μικρή συμμετοχή) δεν ήταν πλήρες και περιείχε πολλά keywords που στην πραγματικότητα ήταν χαμηλής σημασίας τόσο για το άρθρο όσο και για την κατηγορία. Τα πλέον σημαντικότερα αποτελέσματα πηγάζουν από την τρίτη ομάδα χρηστών, τα μέλη της οποίας θεωρούνται από τους πιο 'έμπειρους' στη χρήση του συστήματος με σχεδόν σταθεροποιημένα προφίλ ύστερα από χρήση του συστήματος για μακρύ χρονικό διάστημα. Η σταθερότητα και η πληρότητα του προφίλ των χρηστών αυτών δίνει τη δυνατότητα προσωποποίησης στο μηχανισμό εξαγωγής περίληψης. Τα μέλη αυτής της ομάδας επέλεξαν σε ποσοστό μεγαλύτερο του 90% των άρθρων, την προσωποποιημένη περίληψη ως πιο αντιπροσωπευτική του άρθρου και μόνο 3% των περιλήψεων αξιολογήθηκαν ως 'όμοιες'. Είναι σημαντικό να τονιστεί ότι τα περισσότερα από τα υπολειπόμενα άρθρα (7%), αξιολογήθηκαν από τον μηχανισμό κατηγοριοποίησης του συστήματος ως 'ανήκοντα σε κάποια κατηγορία αλλά με ασθενή συσχέτιση'. Αυτό σημαίνει ότι αυτά ήταν άρθρα τα οποία προστέθηκαν στη συγκεκριμένη κατηγορία με την

Πίνακας 10.2: Αλλαγές στην ακρίβεια και την ανάκληση για την περίληψη ενός άρθρου ύστερα από την προσθήκη πιο αντιπροσωπευτικών *keywords* για την κατηγορία στην οποία το άρθρο ανήκει.

Χρόνος	Άρθρα που προστέθηκαν στην κατηγορία	Υλοποιημένος μηχανισμός	
		Ακρίβεια	Ανάκληση
10 λεπτά	0	0,5	0,66
8 ώρες	8	0,5	0,66
24 ώρες	31	0,66	0,5
36 ώρες	43	0,66	0,66
48 ώρες	59	0,66	0,66
62 ώρες	88	0,75	0,75
78 ώρες	103	0,75	0,8

'υποσημείωση' ότι το σύστημα δεν μπόρεσε με απόλυτη βεβαιότητα να τα κατατάξει σε κάποια κατηγορία, αλλά η κατηγορία στην οποία τελικά εισήχθηκαν είναι η πιο 'κοντινή' για αυτά τα άρθρα.

10.2.3 Αλληλεπίδραση μεταξύ της διαδικασίας περίληψης και κατηγοριοποίησης

Με σκοπό να εκτιμηθεί η αλληλεπίδραση μεταξύ των μηχανισμών περίληψης και κατηγοριοποίησης, διεξάγαμε πειραματική διαδικασία. Για να έχουμε για αρχική βάση γνώσης (ακόμα και μια μικρή), συγκεντρώθηκαν άρθρα νέων από ορισμένα σημαντικά news portals. Ορίσθηκαν 6 διαφορετικές κατηγορίες νέων: business, entertainment, health, politics, science, και sports. Τα κείμενα που κρατήθηκαν, οργανώθηκαν σε αυτές τις κατηγορίες (περίπου 180 σε κάθε μια). Στη συνέχεια, χρησιμοποιώντας τους μηχανισμούς εξαγωγής κειμένου και κατηγοριοποίησης, κρατήθηκε το 50% των keywords για κάθε κείμενο και κάθε keyword συσχετίστηκε με κάθε κατηγορία χρησιμοποιώντας την απόλυτη συχνότητα εμφάνισης ως μέτρο ομοιότητας. Πιο συγκεκριμένα, διεξήχθηκαν τριών ειδών πειραματικές διαδικασίες.

Αρχικά, χρειαζόταν να καθοριστεί το ποσοστό από keywords του κειμένου το οποίο πρέπει να κρατηθεί ουτοσώστε ο μηχανισμός κατηγοριοποίησης να έχει την μεγαλύτερη αποτελεσματικότητα. Προς αυτή την κατεύθυνση, μεταβάλαμε το ποσοστό των keywords που κρατούνται από 0,1 (δηλ. 10% των keywords) σε 1 (δηλ. όλα τα keywords) με βήμα 0,1, κάνοντας χρήση ενός αντιπροσωπευτικού κειμένου για κάθε μια από τις προαναφερθέντες κατηγορίες, και το κατηγοριοποιήσαμε. Το κείμενο που επιλέχθηκε για είσοδο στον μηχανισμό κατηγοριοποίησης δεν ήταν μέρος των κειμένων που χρησιμοποιήθηκαν για την κατασκευή της βάσης γνώσης (δεν ήταν μέρος του training set). Για κάθε ποσοστό από keywords μετρήθηκε η ομοιότητα συνημιτόνου μεταξύ του κειμένου και της κάθε κατηγορίας που υπάρχει στη βάση γνώσης. Εκτελέστηκαν πειράματα χρησιμοποιώντας ελάχιστο μήκος keywords 5 και 6 γράμματα, τόσο για την βάση γνώσης, όσο και για το κείμενο που εισήχθηκε στον μηχανισμό κατηγοριοποίησης. Ακολουθούν ορισμένα διαγράμματα που αποτυπώνουν τα αποτελέσματα.

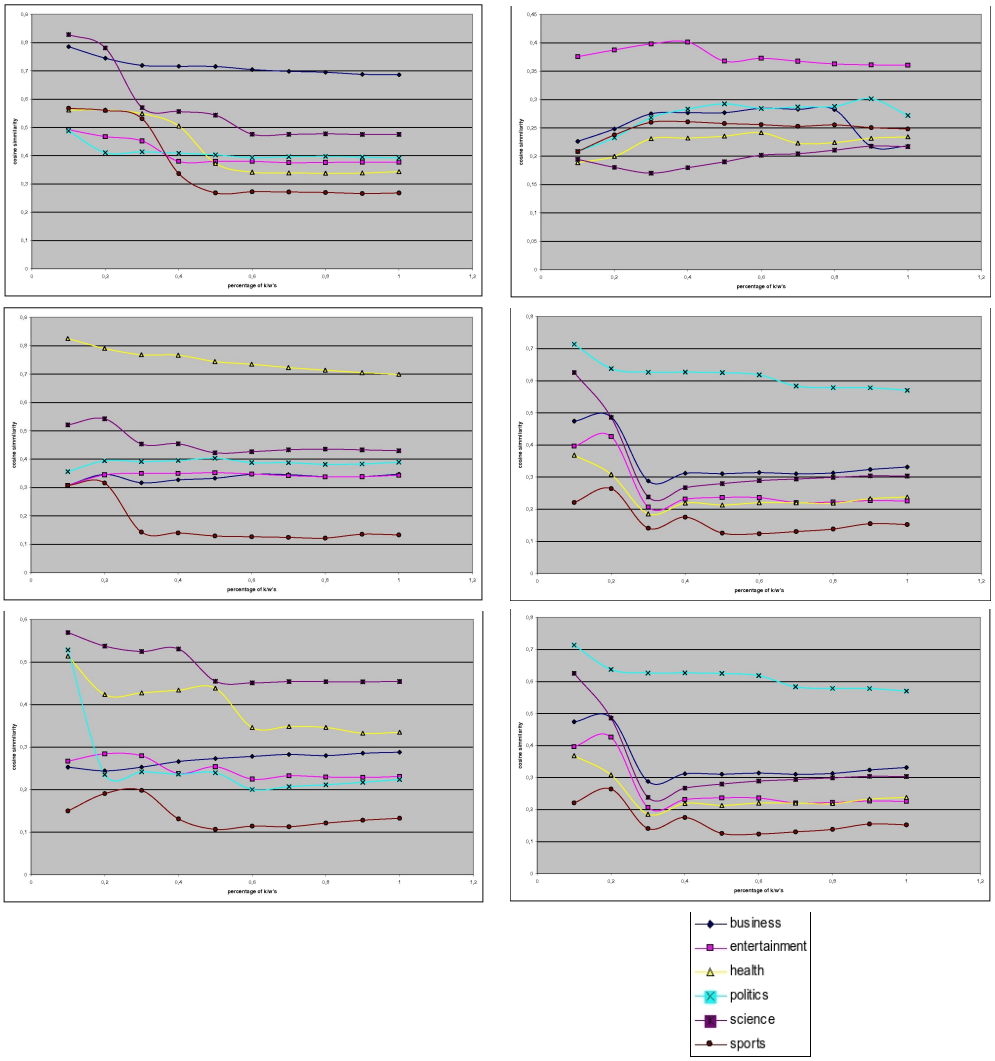
Από το Σχήμα 10.4 (αποτελέσματα διαδικασίας κατηγοριοποίησης), προκύπτει ότι ένα ποσοστό 30% των keywords του κειμένου πρέπει να κρατηθούν από την διαδικασία κατηγοριοποίησης ώστε αυτή να είναι βέλτιστη. Αν και ένα μικρότερο ποσοστό μπορεί να είναι επαρκές ώστε να αποφασιστεί η κατηγορία του κειμένου, κρατάμε ένα ποσοστό 30% διότι, πρώτον μας δίνει σχεδόν πάντα σωστή απόφαση για την κατηγορία του κειμένου και δεύτερον, μας δίνει έναν ισχυρό διαχωρισμό (διαφορά ποσοστού) μεταξύ της σωστής κατηγορίας και των υπολοίπων. Κατά την γνώμη μας, αυτή η διαφορά στην ομοιότητα είναι ο πιο σημαντικός παράγοντας για έναν μηχανισμό κατηγοριοποίησης, αφού μπορεί να μας δώσει σωστές απαντήσεις ακόμη και για μικρή βάση γνώσης. Για παράδειγμα είναι δυνατό, όταν η βάση γνώσης έχει πολλές κατηγορίες μερικές από τις οποίες παρόμοιες, η ομοιότητα μεταξύ ενός κειμένου και παραπάνω από μια κατηγορίες να είναι μεγάλη. Σε αυτή την περίπτωση, η διαφορά στην ομοιότητα μπορεί να είναι ένα καλύτερο μέτρο για την κατηγοριοποίησης, παρά ένα όριο απόλυτης ομοιότητας.

Όπως είναι φανερό από το Σχήμα 10.5, ένα κείμενο μπορεί να επιτύχει καλύτερο σκορ χρησιμοποιώντας ένα ελάχιστο μήκος 5 γραμμάτων για τα keywords και κρατώντας 50% των keyowrds που προκύπτουν. Με αυτό τον τρόπο, η βάση γνώσης είναι πιο φιλτραρισμένη, ενώ δεν μένουν έξω από τη διαδικασία keywords σημαντικά για κάποια/ες κατηγορία/ες.

Στο επόμενο βήμα της πειραματικής διαδικασίας, θέλουμε να εξεταστεί η επιρροή που έχει η διαδικασία περίληψης στο στάδιο της κατηγοριοποίησης. Για να το πετύχουμε αυτό, αρχικά περάστηκαν από το μηχανισμό περίληψης κάποια ανθρωπίνως προκατηγοριοποιημένα κείμενα τα οποία στη συνέχεια προωθήθηκαν στην διαδικασία κατηγοριοποίησης. Τελικά συγκρίναμε την έξοδο του μηχανισμού κατηγοριοποίησης (η οποία με αυτό τον τρόπο μας δίνει την ομοιότητα της περίληψης του κειμένου με τη καταγεγραμμένη κατηγορία που αυτό ανήκει), με την προκαθορισμένη κατηγορία του κειμένου.

Χρησιμοποιήθηκαν διάφορα μεγέθη περιλήψεων με σκοπό να εντοπιστεί η επίδραση που έχουν στην κατηγοριοποίηση της περίληψης. Ακολουθούν ορισμένα διαγράμματα της πειραματικής διαδικασίας χρησιμοποιώντας κείμενα που ανήκουν σε διαφορετικές κατηγορίες, τα οποία αποκαλύπτουν το ιδανικό ποσοστό των προτάσεων οι οποίες μπορούν να διαμορφώσουν μια 'καλή' περίληψη.

Από αυτού του είδους την πειραματική διαδικασία καταλήξαμε στο συμπέρασμα ότι κρατώντας ένα εύλογο μέγεθος από τις αρχικές προτάσεις, περίπου 20%, για την παραγωγή της περίληψης του κειμένου, μπορούμε να κατηγοριοποιήσουμε την περίληψη σωστά στην κατηγορία του κειμένου. Με αυτό τον τρόπο γλιτώνουμε ένα τεράστιο ποσοστό της δουλειάς που πρέπει να γίνει στην πλευρά της κατηγοριοποίησης, αφού η περίληψη

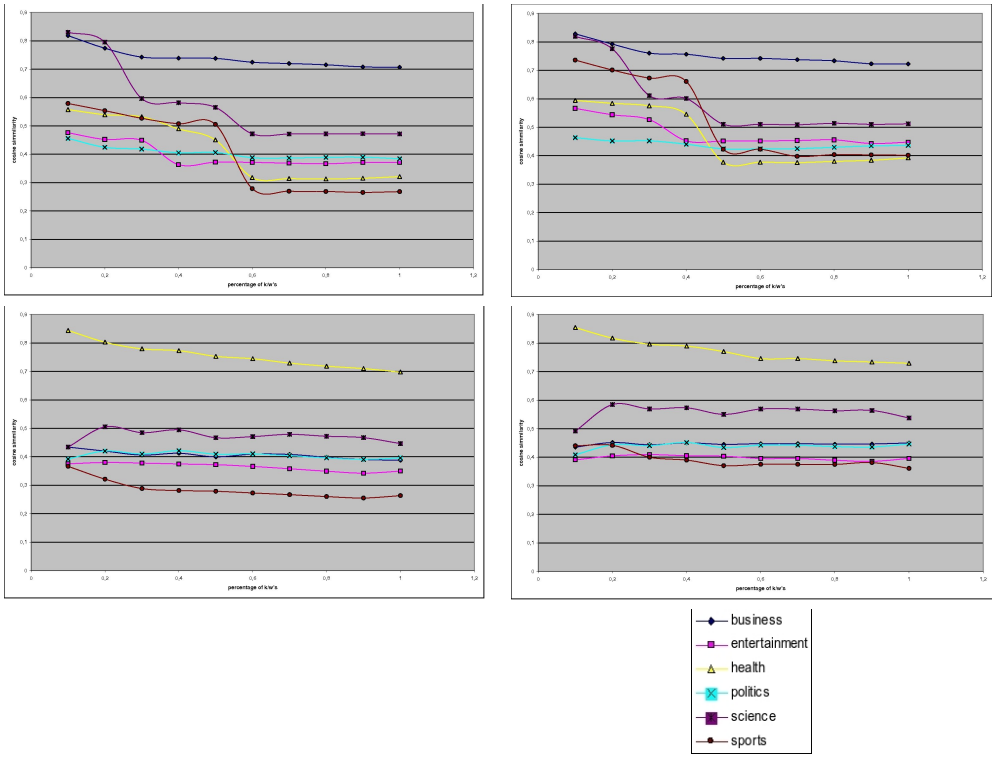


Σχήμα 10.4: Ομοιότητα συνημιτόνου των κειμένων σε σχέση με τις κατηγορίες. Το *Training set* κατασκευάζεται με χρήση του 50% των *keywords* (διαδικασία προεπεξεργασίας).

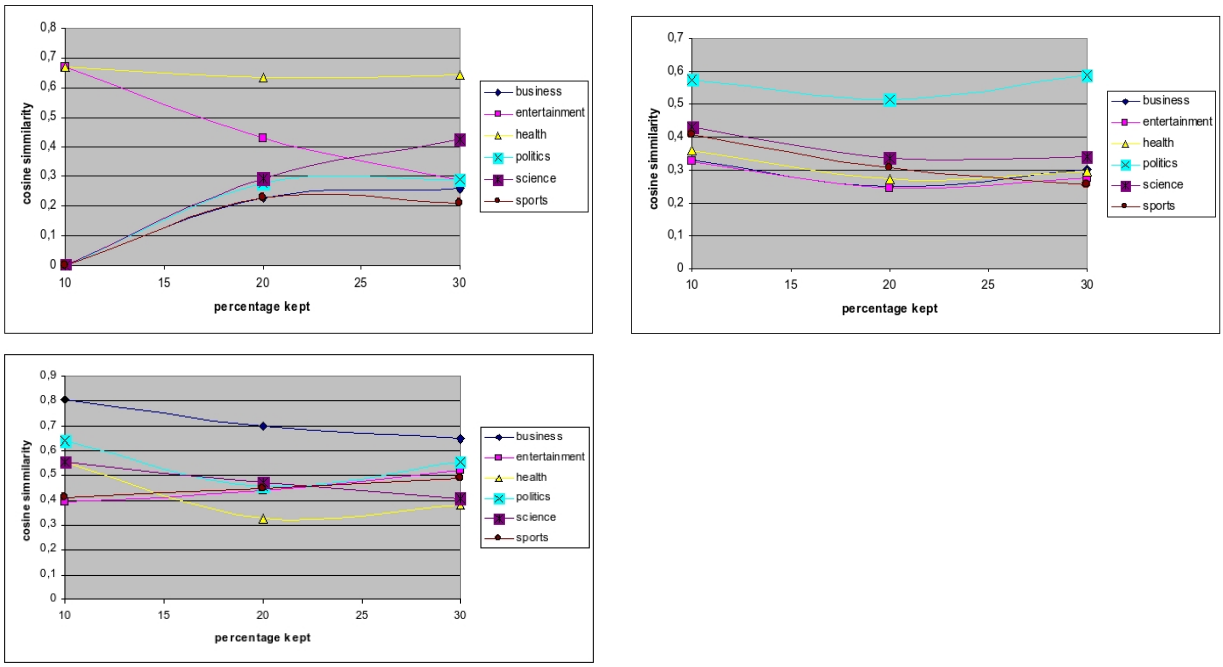
είναι μόνο ένα μικρό μέρος του κειμένου. Αυτό το αποτέλεσμα είναι μεγάλης σημασίας για ένα γρήγορα ανταποκρινόμενο, πραγματικού χρόνου σύστημα κατηγοριοποίησης.

Ένα επιπλέον πεδίο στο οποίο έγινε πειραματισμός αφορούσε τη διερεύνηση της επίπτωσης που έχει η κατηγοριοποίησης στην διαδικασία της περίληψης. Για να αποκαλυφθεί η πιθανή συσχέτιση, κατασκευάσαμε τον μηχανισμό περίληψης ενσωματώνοντας σε αυτόν την δυνατότητα κατηγοριοποίησης. Αυτό σημαίνει πως, όταν γνωρίζουμε εκ' των προτέρων την κατηγορία του κειμένου, μπορούμε να λάβουμε υπ' όψιν αυτή την πληροφορία κατά τη διαδικασία της περίληψης ρυθμίζοντας το βάρος της κάθε πρότασης ανάλογα. Για παράδειγμα, εάν μια πρόταση περιέχει πολλά keywords άσχετα με την κατηγορία του κειμένου (εκ' των προτέρων γνώση), το σκορ της θα είναι πολύ χαμηλό, ή ακόμη και αρνητικό σε σχέση με την περίπτωση που δεν γνωρίζουμε την κατηγορία του κειμένου.

Χρησιμοποιώντας κείμενα από συλλογές κειμένων (corpus texts), αρχικά παρηγάγαμε την περίληψη του κειμένου χωρίς την χρήση του παράγοντα κατηγοριοποίησης (δηλ. k_3=1) και μετά χρησιμοποιήσαμε αυτή την επιπλέον πληροφορία για να παράγουμε μια ακόμη περίληψη. Συγκρίναμε τις δύο περιλήψεις με την 'βέλτιστη' περίληψη που είχαμε από το corpus και που παρήχθη από ανθρώπους. Τα αποτελέσματα είναι αρκετά ενθαρρυντικά αφού βρέθηκε ότι το στοιχείο της κατηγοριοποίησης βελτίωνε τα αποτελέσματα

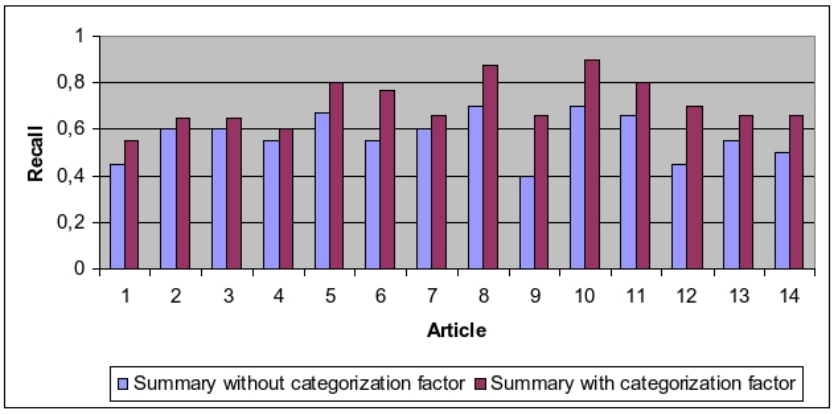


Σχήμα 10.5: Η πρώτη στήλη δείχνει την ομοιότητα συνημιτόνου μετρημένη χρησιμοποιώντας το 50% των *keywords* από το *training set*. Η δεύτερη στήλη δείχνει την ίδια ομοιότητα συνημιτόνου μετρημένη χρησιμοποιώντας το 100% των *keywords* του *training set*.

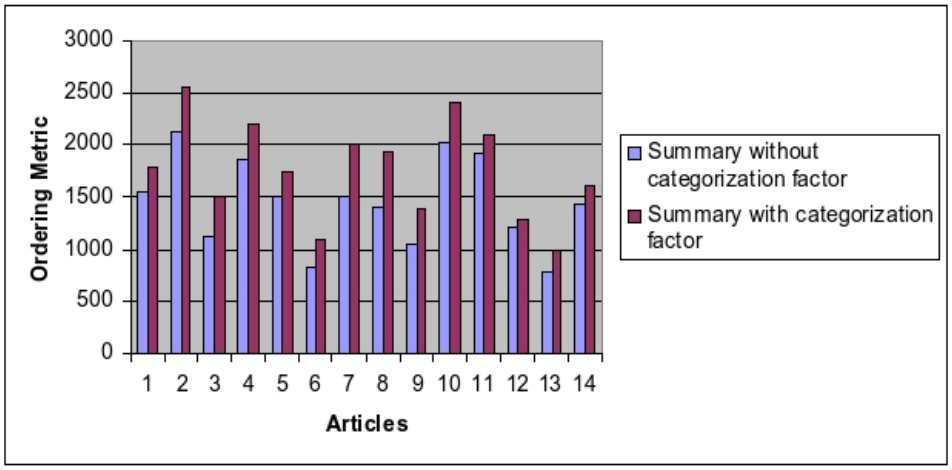


Σχήμα 10.6: Ομοιότητα συνημιτόνου που μετρήθηκε για την κατηγοριοποίηση περιλήψεων χρησιμοποιώντας διάφορα ποσοστά για την δημιουργία των περιλήψεων

της περίληψης κατά περίπου 10% ή ακόμη παραπάνω σε ορισμένες περιπτώσεις, κάτι που σημαίνει ότι οι προτάσεις τις οποίες κράτησε ο μηχανισμός περίληψης μετά τη χρήση της πληροφορίας κατηγοριοποίησης είναι πιο κοντά στις 'βέλτιστες'.



Σχήμα 10.7: Σύγκριση της ανάκλησης των περιλήψεων οι οποίες εξήχθηκαν με και χωρίς την χρήση του παράγοντα κατηγοριοποίησης.



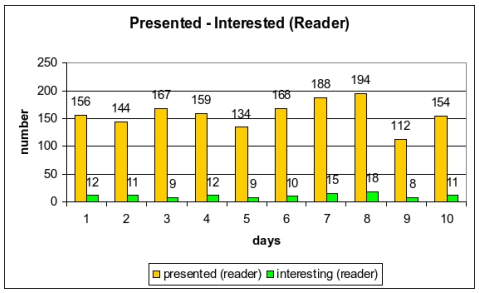
Σχήμα 10.8: Σύγκριση της μετρικής σειράς από περιλήψεις που εξήχθηκαν με και χωρίς τον παράγοντα κατηγοριοποίησης.

Για να συγκρίνουμε τα αποτελέσματα από τις δύο περιπτώσεις (με χρήση της πληροφορίας κατηγοριοποίησης και χωρίς), χρησιμοποιήθηκε η μετρική ανάκλησης, δηλαδή, πόσες από τις προτάσεις της ανθρώπινα εξαγόμενης ('βέλτιστης') περίληψης ανακλήθηκαν από κάθε διαδικασία, και η μετρική σειράς των προτάσεων. Η τελευταία, χρησιμοποιήθηκε για να σημειώσει την σημασία που έχει η σειρά των προτάσεων σε μια περίληψη. Για παράδειγμα, είναι πιθανό και οι δύο τεχνικές περίληψης να επιτύχουν την ίδια ανάκληση προτάσεων αλλά η σειρά των προτάσεων να είναι καλύτερη σε μια από αυτές. Για την ακρίβεια, παρατηρήθηκε ότι η τεχνική περίληψης που κάνει χρήση της πληροφορίας κατηγοριοποίησης επιτυγχάνει όχι μόνο καλύτερη ανάκληση, αλλά και καλύτερη σειρά στις προτάσεις που επιστρέφουν.

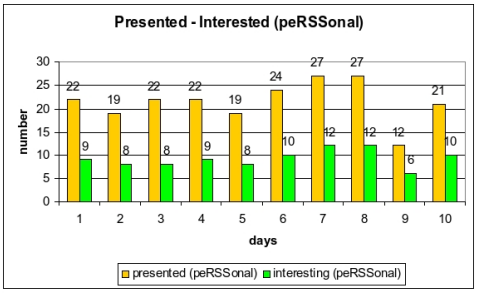
10.3 Σύστημα παρουσίασης

Στην τρέχουσα ενότητα γίνεται μια παρουσίαση και αξιολόγηση του συστήματος peRSSonal. Για να αξιολογηθεί το υποσύστημα παρουσίασης και το κατά πόσο η πληροφορία που φτάνει στο χρήστη είναι

ικανοποιητική, εκτελέστηκε πειραματική διαδικασία. Κατά τη διάρκεια αυτής, δημιουργήθηκαν 10 προφίλ χρηστών με συγκεκριμένες προτιμήσεις σε κατηγορίες νέων, με τα άρθρα από τα οποία τροφοδοτείται ο μηχανισμός (από 10 RSS feeds), να τροφοδοτούνται στους 10 χρήστες. Παράλληλα τροφοδοτείται σε αυτούς και το προσωποποιημένο περιεχόμενο του συστήματος (περίληψη). Αυτό που εξετάστηκε είναι, το κατά πόσον οι χρήστες μπορούν να μείνουν ευχαριστημένοι α) από την επιλογή άρθρων που έγινε γι' αυτούς, και β) από το προσωποποιημένο περιεχόμενο που έχει να κάνει με τα συγκεκριμένα άρθρα που έλαβαν. Επίσης εκτιμάται και η μείωση στο φόρτο των χρηστών στη μία και στην άλλη περίπτωση, σε σχέση με την πληρότητα σε ενημέρωση που μπορούν να έχουν. Σε αυτό το σημείο θα πρέπει να τονιστεί ότι, ναι μεν θέλουμε το σύστημα να κάνει ένα φιλτράρισμα της υπέρογκης πληροφορίας για λογαριασμό των χρηστών, από την άλλη όμως, δεν θέλουμε να χάνονται άρθρα που θεωρούνται σημαντικά από τους χρήστες. Τα αποτελέσματα που προέκυψαν παρουσιάζονται στις γραφικές παραστάσεις του Σχήματος 10.9 και αφορούν ημερήσιες τιμές.



(α)



(β)

Σχήμα 10.9: (α)Τα άρθρα παρουσιάζονται στους χρήστες απ' ευθείας από news portals, (β)Τα άρθρα παρουσιάζονται στους χρήστες από το μηχανισμό

Όπως μπορούμε να δούμε, το σύστημα παρουσιάζει περίπου 85% λιγότερα άρθρα στους χρήστες ημερησίως αλλά το ποσοστό των άρθρων που μοιάζουν ενδιαφέροντα για τους χρήστες είναι πάνω από 40% ενώ στην περίπτωση άμεσης λήψης των άρθρων από news portals το ποσοστό των ενδιαφέροντων άρθρων είναι μόλις 7% των άρθρων που τους παρουσιάζονται. Αυτό σημαίνει ότι ο μηχανισμός μπορεί να επιτύχει καλύτερο 'καθάρισμα' των άρθρων παρέχοντας νέα στους χρήστες που πραγματικά τους ενδιαφέρουν γλιτώνοντάς τους έτσι από τη χρονοβόρα διαδικασία του 'ξεκαθαρίσματος' των άρθρων.

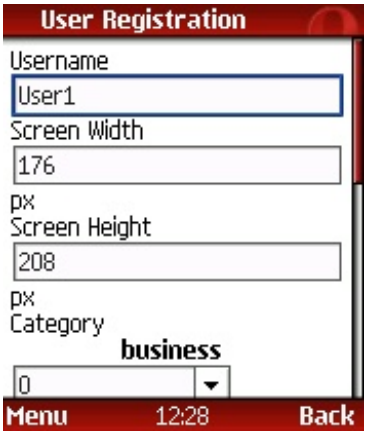
10.3.1 Σύστημα παρουσίασης σε συσκευές μικρού μεγέθους

Σε αυτή την ενότητα γίνεται μια προσπάθεια αξιολόγησης του υποσυστήματος παρουσίασης της προσωποποιημένης περίληψης άρθρων στον τελικό χρήστη συσκευής μικρού μεγέθους. Προς την κατεύθυνση αυτή, παρουσιάζονται κάποια screenshots από τη συσκευή του τελικού χρήστη που χρησιμοποιεί το σύστημα.

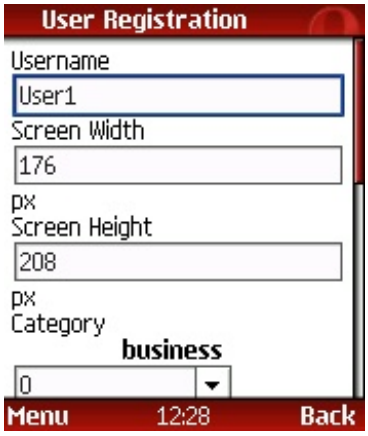
Όταν ένας νέος χρήστης καταφθάνει, οδηγείται στη φόρμα εγγραφής που φαίνεται στο Σχήμα 10.10 όπου εισάγει το όνομά του (ως απαραίτητο στοιχείο ταυτοποίησης) και τις δυνατότητες της συσκευής του (ανάλυση οθόνης). Το δεύτερο ανιχνεύεται αυτόματα από το υποσύστημα παρουσίασης, μπορεί όμως να ορισθεί και από τον χρήστη, και χρησιμοποιείται για τον καθορισμό α) του μήκους των περιλήψεων που στέλνονται στον χρήστη και β) το πλήθος των άρθρων που ταιριάζουν με το μέγεθος της συσκευής του. Ο χρήστης επίσης παρέχει τις προτιμήσεις του για τις κατηγορίες του συστήματος, σε κλίμακα από -5 έως +5.

Όταν ένας μη-εγγεγραμμένος χρήστης ζητάει ένα RSS feed, μια προκαθορισμένη RSS απάντηση, η οποία περιέχει ορισμένες προκαθορισμένες περιλήψεις, στέλνεται πίσω στον χρήστη 10.11. Αντίθετα, αν ο χρήστης είναι ήδη εγγεγραμμένος, του στέλνεται η προσωποποιημένη απάντηση σύμφωνα με το προφίλ του.

Ο σημαντικός παράγοντας που πρέπει να ληφθεί υπ' όψιν, είναι ότι διαφορετικοί χρήστες το συστήματος λαμβάνουν διαφορετικές RSS απαντήσεις που ποικίλουν σε μήκος, σειρά κατάταξης, πλήθος και κατηγορία των νέων και των περιλήψεών τους. Είναι πολύ πιθανό δύο διαφορετικοί χρήστες να λαμβάνουν τα ίδια άρθρα αλλά διαφορετικές περιλήψεις αυτών (βάσει των προτιμήσεων των χρηστών)· αυτή είναι και η περίπτωση του Σχήματος 10.12



(α)



(β)

Σχήμα 10.10: (α)Ο οθόνη εγγραφής στο σύστημα, (β)Επιλογή προτιμήσεων από τον χρήστη

(α)

(β)

Σχήμα 10.11: (α)Μια προκαθορισμένη απάντηση του συστήματος για μη-εγγρεγραμμένο χρήστη, (β)Προσωποποιημένη απάντηση σε εγγεγραμμένο χρήστη

(α)

(β)

Σχήμα 10.12: (α)Απόκριση για τον χρήστη Α σχετικά με ένα άρθρο, (β)Απόκριση για το χρήστη Β για το ίδιο άρθρο

ΚΕΦΑΛΑΙΟ 11

Συμπεράσματα και μελλοντική εργασία

> Ignorance, the root and the stem of every evil.
>
> *Plato, Greek author & philosopher*

Το Διαδίκτυο έχει λάβει χαοτικές διαστάσεις και η πληροφορία που διακινείται σε αυτό είναι υπέρογκη. Στην εποχή μας και με τα μέσα που διαθέτει ακόμα και ο απλός χρήστης, η προσθήκη περιεχομένου στο Διαδίκτυο από τον καθένα, είναι μια διαδικασία το ίδιο εύκολη με την απλή περιαγωγή στο χώρο του παγκόσμιου ιστού (π. χ. φαινόμενο blogging, το Web 2.0, κ.ο.κ.). Το πρόβλημα που δημιουργεί αυτή η ανεξέλεγκτη κατάσταση είναι ότι ακόμα οι πιο έμπειροι χρήστες καταναλώνουν πολύ χρόνο στην προσπάθεια εύρεσης πληροφορίας και συγκεκριμένα πηγών ενημέρωσης για τα θέματα που τους ενδιαφέρουν.

Εστιασμένοι στο πρόβλημα που περιγράφεται στην προηγούμενη παράγραφο, επικεντρωνόμαστε στην πληροφορία που διακινείται στο διαδίκτυο και αφορά νέα και γεγονότα. Αυτό που θέλουμε ουσιαστικά να δημιουργήσουμε είναι ένα σύστημα το οποίο θα είναι σε θέση να παρουσιάζει, ειδήσεις που δημοσιεύονται στο Διαδίκτυο, με τρόπο απλό και έχοντας στο νου μας τον παράγοντα άνθρωπο. Για να το επιτύχουμε αυτό πρέπει να παρέχουμε στο χρήστη μία δικτυακή υπηρεσία η οποία θα μπορεί να προσαρμόζεται σε αυτόν και να του παρέχει ποιοτικό και πλήρες περιεχόμενο για τις εξελίξεις που τον ενδιαφέρουν. Δεν στοχεύουμε στην ανάπτυξη ενός ακόμα portal νέων αφού κάτι τέτοιο δεν θα αντιμετώπιζε το πρόβλημα.

Το σύστημα που δημιουργήθηκε στα πλαίσια της διπλωματικής εργασίας περνάει την πληροφορία μέσα από διάφορα στάδια επεξεργασίας. Αρχικά γίνεται το διαπέρασμα των ιστοσελίδων γνωστών news portals και αποθηκεύεται ο html κώδικάς τους. Στη συνέχεια, από τον html κώδικα αναγνωρίζεται η χρήσιμη πληροφορία που αφορά το συγκεκριμένο άρθρο. Είναι σημαντικό σε αυτό το σημείο να απομακρύνονται όσο το δυνατόν περισσότερα περιττά στοιχεία της σελίδας και να κρατούνται μόνο τα απαραίτητα. Ο μηχανισμός που αναπτύχθηκε γι΄ αυτό το φιλτράρισμα, βασίζεται στους αλγορίθμους που περιγράφηκαν σε προηγούμενα κεφάλαια και κάνει αρκετά καλή δουλεία. Έχει όμως αρκετά περιθώρια βελτίωσης ώστε τόσο να απομακρύνει επιβαρυντικά στοιχεία της σελίδας για το μηχανισμό (π. χ. ορισμένα scripts καταφέρνουν να περνάνε από τη διαδικασία), όσο και να μην χάνει σημαντικά τμήματα του άρθρου θεωρώντας τα ως άχρηστη πληροφορία. Σημαντικό ρόλο σε αυτό παίζει και η διαμόρφωση της σελίδας που χρησιμοποιούν τα news portal καθώς αν αυτή είναι εντελώς άναρχη και δίχως δομή είναι προφανές ότι η δουλειά του φιλτραρίσματος γίνεται δυσκολότερη.

Ακολουθώντας το φιλτράρισμα του κειμένου ο μηχανισμός προχωρά με τη διαδικασία της προεπεξεργασίας κειμένου και εξαγωγής των κωδικολέξεων. Πρόκειται για τη θεμελιώδη διεργασία σχεδόν όλων των μηχανισμών ανάκτησης πληροφορίας και επομένως επιθυμούμε τα καλύτερα δυνατά αποτελέσματα. Ο μηχανισμός προεπεξεργασίας που κατασκευάστηκε δοκιμάστηκε διεξοδικά ώστε να παράγει σωστές εξόδους και να κρατά τον πλεονασμό σε χαμηλά επίπεδα. Περιλαμβάνει τις διαδικασίες αφαίρεσης σημείων στίξης και αριθμών, αφαίρεσης των λέξεων που ανήκουν στη λίστα των stopwords, το stemming των λέξεων και φυσικά την αντιστοίχηση των keywords που προκύπτουν με τις προτάσεις όπου αυτά εμφανίζονται. Η όλη

διαδικασία κάνει εκτεταμένη χρήση regular expressions της βιβλιοθήκης boost-regex της C++ και είναι υλοποιημένη ώστε να αποφεύγονται περιττοί έλεγχοι ή επανάληψης με στόχο τη βελτίωση της απόδοσης.

Μπορεί από προγραμματιστική άποψη το τμήμα της εξαγωγής κωδικολέξεων να δέχεται λίγες βελτιώσεις, είναι δυνατή όμως η περαιτέρω βελτίωση του υποσυστήματος βελτιώνοντας τα παρακάτω σημεία:

- Χρήση μιας καλύτερης λίστας από stopwords η οποία ούτε θα απορρίπτει σημαντικές λέξεις ούτε και θα δέχεται άλλες μη χρήσιμες. Μπορεί π. χ. να χρησιμοποιηθεί μια δυναμική λίστα που προσαρμόζεται ανάλογα με τη θεματολογία του κειμένου.

- Χρήση ενός καλύτερου stemmer που βασίζεται σε κανόνες και έχει ελάχιστα λάθη σε αντίθεση με τον porter stemmer που βασίζεται ελάχιστα σε κανόνες και περισσότερο σε 'έτοιμες' καταλήξεις.

- Χρήση λεξικών για ορθογραφικό έλεγχο του κειμένου προτού λανθασμένες λέξεις αξιολογηθούν ως keywords.

- Εντοπισμός ουσιαστικών από το κείμενο καθώς τα ουσιαστικά είναι ως επί το πλείστον οι λέξεις με τη μεγαλύτερη βαρύτητα (από άποψη νοήματος) μέσα σε μία πρόταση. Επιπρόσθετα, ανανέωση της βάσης γνώσης ώστε να περιέχει μόνο ουσιαστικά.

- Αναγνώριση γλώσσας κειμένου και ουσιαστικά επέκταση του όλου μηχανισμού σε ένα ενοποιημένο πολυγλωσσικό περιβάλλον τεχνικών ανάκτησης πληροφορίας.

Όσον αφορά στην μετατροπή του μηχανισμού σε πολυγλωσσικό σύστημα επεξεργασίας κειμένων, αυτό θα πρέπει να περιλαμβάνει λίστες με stopwords σε διάφορες γλώσσες, κανόνες stemming για την εκάστοτε γλώσσα καθώς και τα λεξικά των γλωσσών. Οι πληροφορίες αυτές θα πρέπει να αποθηκεύονται κεντρικά στη ΒΔ και να είναι διαθέσιμες on the fly κατά την εκτέλεση του μηχανισμού και αφού αναγνωριστεί η γλώσσα του κειμένου. Σε αρχική φάση εστιαζόμαστε στην επέκταση του μηχανισμού για την ελληνική γλώσσα.

Όσον αφορά στα υποσυστήματα κατηγοριοποίησης και αυτόματης εξαγωγής περίληψης, στοχεύουμε σε ανάπτυξη πολλαπλών διαφορετικών αλγορίθμων μια και αυτά τα υποσυστήματα είναι ανεξάρτητα από τον υπόλοιπο μηχανισμό (modules). Επίσης είναι εφικτή η ανάπτυξη γραφικού περιβάλλοντος για τον όλο μηχανισμό με στόχο την εύκολη πρόσβαση και διαχείρισή του. Παράλληλα, η ολοκλήρωση των υποσυστημάτων λήψης και φιλτραρίσματος πληροφορίας σε γλώσσα C++ θα προσφέρει ταχύτητα και ομοιογένεια στον μηχανισμό, τα υπόλοιπα μέρη του οποίου είναι σε C++.

Το υποσύστημα που αφορά τις συσκευές μικρού μεγέθους μπορεί να βελτιωθεί λαμβάνοντας υπ' όψιν του επιπλέον παράγοντες πέρα από την ανάλυση της συσκευής του χρήστη, π. χ. δυνατότητα δικτύωσης, δυνατότητες για browsing, κ.ο.κ. Ο χρήστης θα μπορεί να εισάγει μόνο το όνομα της συσκευής τους και τα υπόλοιπα στοιχεία θα είναι διαθέσιμα στον μηχανισμό μέσω στοιχείων αποθηκευμένων στη ΒΔ ή μέσω on-line στοιχείων.

Ευρετήριο

Βιβλιογραφία

[1] amazon.com. Online shopping. http://www.amazon.com.

[2] The apache web server. Website. http://httpd.apache.org/.

[3] Bbc news. News portal. http://news.bbc.co.uk/.

[4] Boost libraries for c++. Website. http://www.boost.org/.

[5] Breaking news, world, u.s., video, investing and business news & more — reuters.com. News portal. http://www.reuters.com/.

[6] Cgicc. a c++ class library for writing cgi applications. Website. http://www.cgicc.org/.

[7] Cnn.com - breaking news, u.s., world, weather, entertainment & video news. News portal. http://www.cnn.com/.

[8] Dick hardt. how sxip works (whitepaper). Website. https://sxip.org/docs/specs/how-sxip-works.pdf2004.

[9] foxnews.com. News portal. http://www.foxnews.com/.

[10] gartner.com. Website. http://www.gartner.com/.

[11] The gnu compiler collection. Website. http://www.netbeans.org/.

[12] Gnu wget - gnu project - free software foundation. Website. http://www.gnu.org/software/wget/.

[13] google.com. Search engine. http://www.google.com/.

[14] Heritrix internet archive's open-source, extensible, web-scale, archival-quality web crawler project. Website. http://crawler.archive.org/.

[15] ht://dig internet search engine software. Website. http://www.htdig.org/.

[16] Httrack website copier - offline browser. Website. http://www.httrack.com/.

[17] Java, java history. http://ils.unc.edu/blaze/java/javahist.html.

[18] The java language. Website. http://www.java.com/.

[19] Kdevelop. integrated development environment for unix, supporting kde/qt, c/c++ and many other languages. Website. http://www.kdevelop.org/.

[20] Larbin web crawler. Website. http://larbin.sourceforge.net/index-eng.html.

[21] Methabot web crawler. Website. http://bithack.se/methabot/.

[22] Mysq+. c++ api interface to the mysql database. Website. http://www.mysql.org/downloads/api-mysql++.html.

[23] Mysql, opensource database. http://www.mysql.com.

[24] Netbeans ide for java. Website. http://gcc.gnu.org/.

[25] Nutch open source web search engine. Website. http://lucene.apache.org/nutch/.

[26] Open directory project. Website. http://www.dmoz.org.

[27] passport.net. Website. http://www.passport.net.

[28] The php language runtime engine: Cli, cgi and apache2 sapis. Website. http://www.php.net/.

[29] The porter stemmer algorithm. Website. http://www.tartarus.org/~martin/PorterStemmer/.

[30] Postgresql, opensource database. http://www.postrgresql.org.

[31] Rss - real simple syndication. Website. http://www.w3.org/WAI/highlights/about-rss.html.

[32] W3c - xml protocol. Website. http://www.w3.org/XML/.

[33] Web information retrieval environment (wire). Website. http://www.cwr.cl/projects/WIRE/.

[34] Websphinx: A personal, customizable web crawler. Website. http://www.cs.cmu.edu/~rcm/websphinx/.

[35] Www size. Website. http://www.worldwidewebsize.com/.

[36] yahoo.com. Search engine. http://www.yahoo.com/.

[37] J. Allan, H. Jin, M. Rajman, C. Wayne, D. Gildea, V. Lavrenko, R. Hoberman, and D. Caputo. Topic-based novelty detection: 1999 summer workshop at CLSP, final report, 1999.

[38] C. Apté, F. Damerau, and S.M. Weiss. *Towards language independent automated learning of text categorization models*. Springer-Verlag New York, Inc. New York, NY, USA, 1994.

[39] H. Arimura, A. Wataki, R. Fujino, and S. Arikawa. A fast algorithm for discovering optimal string patterns in large text databases. *Proc. the 8th International Workshop on Algorithmic Learning Theory*, 1501:247–261.

[40] R. Barzilay and L. Lee. Catching the drift: Probabilistic content models, with applications to generation and summarization. *Proceedings of HLT-NAACL 2004*, pages 113–120, 2004.

[41] N.J. Belkin and W.B. Croft. Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12):29–38, 1992.

[42] VD Belur. Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques. *IEEE Computer Society Press, New York: IEEE press*, 1991.

[43] H. Berger and D. Merkl. A Comparison of Text-Categorization Methods applied to N-Gram Frequency Statistics. *Proc. of the 17th Australian Joint Conf. on Artificial Intelligence*, 2004.

[44] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic Web. *Scientific American*, 284(5):28–37, 2001.

[45] P. Boldi, B. Codenotti, M. Santini, and S. Vigna. UbiCrawler: a scalable fully distributed Web crawler. *Software- Practice and Experience*, 34(8):711–726, 2004.

[46] R. Brandow, K. Mitze, and L.F. Rau. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management: an International Journal*, 31(5):675–685, 1995.

[47] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[48] G. Cao. Support Vector Machine Active Learning with Applications to Text Classification.

[49] M.F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text Databases and Document Management: Theory and Practice*, pages 78–102, 2001.

[50] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan. Using taxonomy, discriminants, and signatures for navigating in text databases. *Proceedings of the 23rd VLDB Conference*, pages 446–455, 1997.

[51] P.K. Chan. Constructing Web User Profiles: A non-invasive Learning Approach. *KDD-99 Workshop on Web Usage Analysis and User Profiling*, pages 7–12, 1999.

[52] W.W. Cohen. Text categorization and relational learning. *Proceedings of ICML-95, 12th International Conference on Machine Learning*, pages 124–132, 1995.

[53] N. Collier, C. Nobata, and J. Tsujii. Extracting the names of genes and gene products with a hidden Markov model. *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 201–207, 2000.

[54] Colleen E. Crangle. Text summarization in data mining. In *Soft-Ware 2002: Proceedings of the First International Conference on Computing in an Imperfect World*, pages 332–347, London, UK, 2002. Springer-Verlag.

[55] L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall, and J. Reagle. The Platform for Privacy Preferences 1.0 (P3P1. 0) Specification. *W3C Recommendation*, 16, 2002.

[56] R.L. Donaway, K.W. Drummey, and L.A. Mather. A comparison of rankings produced by summarization evaluation measures. *ANLP/NAACL Workshops*, pages 69–78, 2000.

[57] S. Dumais and H. Chen. Hierarchical classification of Web content. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263, 2000.

[58] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. *Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155, 1998.

[59] J. Edwards, K. McCurley, and J. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. *Proceedings of the 10th international conference on World Wide Web*, pages 106–113, 2001.

[60] B. Endres-Niggemeyer et al. *Summarizing information*. Springer New York, 1998.

[61] R. Evans, R. Gaizauskas, L. Cahill, J. Walker, J. Richardson, and A. Dixon. POETIC: a system for gathering and disseminating traffic information. *Journal of Natural Language Engineering*, 1(4), 1995.

[62] T. Firmin and M.J. Chrzanowski. An Evaluation of Automatic Text Summarization Systems. *Advances in Automatic Text Summarization*, pages 325–336, 1999.

[63] W.B. Frakes and R. Baeza-Yates. *Information retrieval: data structures and algorithms*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1992.

[64] W. Frawley, G. Piatesky-Shapiro, and C. Matheus. An Overview In Knowledge Discovery in Databases AAAI, 1991.

[65] J.C. French, A.L. Powell, J. Callan, C.L. Viles, T. Emmitt, K.J. Prey, and Y. Mou. Comparing the performance of database selection algorithms. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 238–245, 1999.

[66] D. Fum, G. Guida, and C. Tasso. Forward and backward reasoning in automatic abstracting. *Proceedings of the 9th conference on Computational linguistics-Volume 1*, pages 83–88, 1982.

[67] J. Furnkranz, T. Mitchell, and E. Riloff. A case study in using linguistic phrases for text categorization on the WWW. *Learning for Text Categorization: Proceedings of the 1998 AAAI/ICML Workshop*, pages 98–05, 1998.

[68] J. Goecks and J. Shavlik. Automatically Labeling Web Pages Based on Normal User Actions. *Proc. of the Intl. Conf. on Intelligent User Interfaces*.

[69] J. Goldstein, M. Kantrowitz, V. Mittal, and J. Carbonell. Summarizing text documents: sentence selection and evaluation metrics. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 121–128, 1999.

[70] U. Hahn and U. Reimer. Semantic Parsing and Summarizing of Technical Texts in the TOPIC System. *Informationslinguistik*, pages 153–193, 1986.

[71] P. Hensley, M. Metral, U. Shardanand, D. Converse, and M. Myers. Proposal for an Open Profiling Standard. *Technical Note, World Wide Web Consortium, June*, 1997.

[72] A. Heydon and M. Najork. Mercator: A scalable, extensible Web crawler. *World Wide Web*, 2(4):219–229, 1999.

[73] K. Hoang and P. Do. Discovering Motiv Based Association Rules in a Set of DNA sequences. *RSCTC*, pages 386–390, 2000.

[74] PS Jacobs and L.F. Rau. SCISOR: extracting information from on-line news. *Communications of the ACM*, 33(11):88–97, 1990.

[75] C. Jacquemin. *Spotting and Discovering Terms Through Natural Language Processing*. MIT Press, 2001.

[76] T. Joachims. *Text Categorization with Suport Vector Machines: Learning with Many Relevant Features*. Springer-Verlag London, UK, 1998.

[77] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A Tour Guide for the World Wide Web. *Proceedings of IJCAI97*, pages 1–7, 1997.

[78] K.S. Jones. Exhaustivity and specificity. *Journal of Documentation*, 28(1):11–21, 1972.

[79] M. Jones, G. Marsden, N. Mohd-Nasir, K. Boone, and G. Buchanan. Improving Web interaction on small displays. *COMPUT. NETWORKS*, 31(11):1129–1137, 1999.

[80] J. Kazama, T. Makino, Y. Ohta, and J. Tsujii. Tuning Support Vector Machines for Biomedical Named Entity Recognition. *Proc. of the Workshop on Natural Language Processing in the Biomedical Domain (at ACL'2002)*, pages 1–8, 2002.

[81] G. Klyne, F. Reynolds, C. Woodrow, H. Ohto, J. Hjelm, M. Butler, L. Tran, et al. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. *W3C Working Draft*, 8, 2002.

[82] K. Knight and D. Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, 2002.

[83] Y. Koike, T. Kamba, and M. Langheinrich. PIDL-Personalized Information Description Language. *W3C Note, 09*.

[84] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 170–178, 1997.

[85] R. Krovetz. Viewing morphology as an inference process. *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202, 1993.

[86] M. Lennon. Pierce. D., Tarry, B.. & Willett, P.(198 1). *An evaluation of the stemming algorithms.*

[87] D.D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. *Third Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, 1994.

[88] J.B. Lovins. Development of a Stemming Algorithm. 1968.

[89] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, Vol. 2:159–165, 1958.

[90] I. Mani and M. Maybury. *Advances in automatic text summarization.* The MIT Press, 1999.

[91] I. Mani and G. Wilson. Robust temporal processing of news. *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 69–76, 2000.

[92] D. Marcu. The rhetorical parsing of natural language texts. *Proceedings of the 35th annual meeting on Association for Computational Linguistics*, pages 96–103, 1997.

[93] B. Masand. Lino, G., & Waltz, D.(1992). Classifying news stories using memory based reasoning. *Proceedings of 506 15th ACM SIGIR international conference on research and development in information retrieval*, pages 59–65.

[94] L.A. Mather and J. Note. Discovering Encyclopedic Structure and Topics in Text. *Sixth ACM SIGKDD.*

[95] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. *AAAI-98 Workshop on Learning for Text Categorization*, 752, 1998.

[96] T. Mitchell, B. Buchanan, G. DeJong, T. Dietterich, P. Rosenbloom, and A. Waibel. Machine Learning. *Annual Review of Computer Science*, 4(1):417–433, 1990.

[97] D. Mladenic and M. Grobelnik. Word sequences as features in text-learning. *Proceedings of ERK-98, the Seventh Electrotechnical and Computer Science Conference*, pages 145–148, 1998.

[98] B. Mobasher, R. Cooley, and J. Srivastava. Automatic Personalization Through Web Usage Mining. 2000.

[99] MC Mont, S. Pearson, and P. Bramhall. Towards accountable management of identity and privacy: sticky policies and enforceable tracing services. *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, pages 377–382, 2003.

[100] M. Montes-y Gómez, A. Gelbukh, and A. López-López. Mining the News: Trends, Associations, and Deviations. *Computación y Sistemas*, 5(1):14–24, 2001.

[101] C. Mooers. Information retrieval viewed as temporal signalling. *International Congress of Mathematicians. Cambridge, Mass., 1950. Proceedings*, 1951.

[102] A.H. Morris, G.M. Kasper, and D.A. Adams. The effects and limitations of automated text condensing on reading comprehension performance. *Information Systems Research*, 3(1):17–35, 1992.

[103] H.T. Ng, W.B. Goh, and K.L. Low. Feature selection, perception learning, and a usability case study for text categorization. *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 67–73, 1997.

[104] C. Nobata, N. Collier, and J. Tsujii. Automatic term identification and classification in biology texts. *Proc. of the 5th NLPRS*, pages 369–374, 1999.

[105] M. Oka and Y. Ueda. Evaluation of Phrase-representation Summarization based on Information Retrieval Task. *Proceedings of the Workshop on Automatic Summarization, ANLP-NAACL2000*, 2000.

[106] M.E. Okurowski, H. Wilson, J. Urbina, T. Taylor, R.C. Clark, and F. Krapcho. Text summarizer in use: Lessons learned from real world deployment and evaluation. *Proceedings of the Workshop on Automatic Summarization, ANLP-NAACL2000*, 2000.

[107] C.D. Paice. Another stemmer. *ACM SIGIR Forum*, 24(3):56–61, 1990.

[108] CD Paice. Constructing literature abstracts by computer: techniques and prospects. *Information Processing and Management: an International Journal*, 26(1):171–186, 1990.

[109] M. Pazzani, J. Muramatsu, and D. Billsus. Syskill & Webert: Identifying interesting web sites. *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 5461, 1996.

[110] B. Pinkerton. Finding What People Want: Experiences with the WebCrawler. *Proceedings of the Second International World Wide Web Conference*, 1994.

[111] M. Porter. The Porter Stemming Algorithm. *Accessible at http://www. tartarus. org/martin/-PorterStemmer*.

[112] GJ Rath, A. Resnick, and TR Savage. The formation of abstracts by the selection of sentences. *American Documentation*, 12(2):139–143, 1961.

[113] PC Reghu Raj and S. Raman. Content identification and semantic indexing of text documents. *Proc. of the Indo European Conference on Multilingual Communication Technologies (IEMCT-02)*, pages 203–217, 2002.

[114] E. Riloff and J. Shepherd. A corpus-based approach for building semantic lexicons. *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124, 1997.

[115] J. Rissanen. *Stochastic Complexity in Statistical Inquiry Theory*. World Scientific Publishing Co., Inc. River Edge, NJ, USA, 1989.

[116] H. Saggion and G. Lapalme. Concept identification and presentation in the context of technical text summarization. *ANLP/NAACL Workshops*, pages 1–10, 2000.

[117] G. Salton, J. Allan, C. Buckley, and A. Singhal. Automatic Analysis, Theme Generation, and Summarization of Machine-Readable Texts. *Science*, 264(5164):1421, 1994.

[118] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc. New York, NY, USA, 1986.

[119] B. Sankaran. Tamil Search Engine.

[120] M. Saravanan and S. Raman. The term distribution model for summarization of multiple documents. *Proceedings of the Indo European Conference on Multilingual Communication Technologies (IEMCT 2002)*, pages 182–192, 2002.

[121] M. Saravanan, Pc Reghu Raj, and S. Raman. Summarization and Categorization of text data in high-level data cleaning for information retrieval. *Applied Artificial Intelligence*, 17(5):461–474, 2003.

[122] R.C. Schank. *Reading and Understanding: Teaching from the Perspective of Artificial Intelligence*. Lawrence Erlbaum Associates, 1982.

[123] J. Shavlik, S. Calcari, T. Eliassi-Rad, and J. Solock. An instructable, adaptive interface for discovering and monitoring information on the World-Wide Web. *Proceedings of the 4th international conference on Intelligent user interfaces*, pages 157–160, 1998.

[124] N. Slonim and N. Tishby. The power of word clusters for text classification. *Proceedings of ECIR-01, 23rd European Colloquium on Information Retrieval Research*, 2001.

[125] R. Swan and J. Allan. Automatic generation of overview timelines. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, 2000.

[126] K. Tzeras and S. Hartmann. Automatic indexing based on Bayesian inference networks. *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 22–35, 1993.

[127] V.N. Vapnik. *Statistical learning theory*. Wiley New York, 1998.

[128] J. Verbeek. An information theoretic approach to finding word groups for text classification. *Institute for Lang uage, L og ic and Comp utation, University of Am sterdam*, 2000.

[129] D.H. Widyantoro, T.R. Ioerger, and J. Yen. Learning user interest dynamics with a three-descriptor representation. *Journal of the American Society for Information Science and Technology*, 52(3):212–225, 2001.

[130] WA Woods and JG Schmolze. The KL-ONE family. Semantic Networks in Artificial Intelligence, Pp133-178, 1992.

[131] Y. Yang. Expert network: effective and efficient learning from human decisions in text categorization and retrieval. *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 13–22, 1994.

[132] Y. Yang and C.G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems (TOIS)*, 12(3):252–277, 1994.

[133] Y. Yang and J.O. Pedersen. A comparative study on feature selection in text categorization. *Proceedings of the Fourteenth International Conference on Machine Learning*, 97, 1997.

[134] S.R. Young and P.J. Hayes. Automatic classification and summarization of banking telexes. *Proceedings of the Second Conference on Artificial Intelligence Applications*, pages 402–408, 1985.

[135] D. Zeinalipour-Yazti and M. Dikaiakos. Design and implementation of a distributed crawler and filtering processor. *Proc. of NGITS 2002*, 2382:58–74.

ΠΑΡΑΡΤΗΜΑ A

Σημαντικά τμήματα κώδικα

Δεδομένου ότι οι συνολικές γραμμές κώδικα του συστήματος που προέκυψε ξεπερνούν τις 5000, παραθέτουμε στη συνέχεια μόνο τα σημαντικότερα τμήματα αυτού τα οποία αφορούν στην κατηγοριοποίηση, στην περίληψη κειμένου, καθώς και στη κεντρική ροή διεργασιών του μηχανισμού σε C++.

A.1 Πυρήνας κώδικα κατηγοριοποίησης

```
/**
Function:       categorize
Input:          filename of input XML-structured text for the categorization
process, minimum word size (e.g. 4 letters) to keep, training set flag,
percentage of keywords that sould be kept, user id, article id
Operation:      categorizes the input text to the predefined categories with a
relativity measure. If training flag is set, the k/w extraction process takes
place and, given the text's type (category) given with the XML-structured input
text, the text's k/ws are added to the dynamically changing training set of
texts residing in the database
Returns:        t_art_CATEGORIZED, if the text could be categorized
                t_art_UNCATEGORIZED, if the text could not be categorized
(belongs to more than 1 category)
                t_art_TRAINING, if the text could be categorized and should be
added to the training set since it belongs with high relevance to one category.
**/



t_art_STATUS mechanism::categorize(char * filename,int word_size,int
training_set,int kw_percentage, int user_id, int ar_id, map<double,string>
*cosine_simmilarities_sorted){
....
....
....

        if (training_set==1) { //If this text is aimed to be a training set for
the database
                //      /*Check for the text's category, if it already exists,
dont create it*/
                //
                /*First insert into the articles_training the data*/
                mysqlpp::Query query = con.query();

                query << "select * from mechanism_training.category where
cat_title='"+ lol.getType()+"'";
#ifdef SHOW_QUERIES
                clog<<"Query is:"<<query.preview()<<endl;
#endif
                mysqlpp::Result res = query.store();
                query.reset();
                total_queries++;
                int cat_id;

//      clog<<"Total records: "<<res.size()<<endl;
```

```
                if (res.size()==1){
                        //clog<<"Category already exist"<<endl;
                        mysqlpp::Row row;
                        row = res.fetch_row();
                        std::istringstream ss(row["cat_id"].get_string());
                        ss >> cat_id;
//                      clog<<"Category id is:"<<cat_id<<endl;
                }
                else if (res.size()!=0){
                        clog<<"Please check category structure, internal
error!"<<endl;
                        exit(-5);
                }
                else{
                //Creating category first
                        query << "INSERT INTO mechanism_training.category
(cat_title) values ('"+lol.getType()+"')";
#ifdef SHOW_QUERIES
                        clog<<"Query is:"<<query.preview()<<endl;
#endif
                        query.execute();
                        query.reset();
                        total_queries++;
                //Now ask what is the new category's id
                        query << "select * from category where cat_title='"+
lol.getType()+"'";
#ifdef SHOW_QUERIES
                        clog<<"Query is:"<<query.preview()<<endl;
#endif
                        mysqlpp::Result res2 = query.store();
                        query.reset();
                        total_queries++;
                        mysqlpp::Row row;
                        row = res2.fetch_row();
                        std::istringstream ss(row["cat_id"].get_string());
                        ss >> cat_id;
//                      clog<<"Category id is:"<<cat_id<<endl;

                }



                /*Find out the data from the articles table*/
                query << "select ar_title, ar_body, ar_url, ar_date from
mechanism_training.articles_training where
ar_title='"<<escape<<lol.getTitle()<<"'";
/*#ifdef SHOW_QUERIES*/
                clog<<"Query is:"<<query.preview()<<endl;
// #endif
                res = query.store();
                query.reset();
                total_queries++;
                mysqlpp::Row row;
                row = res.fetch_row();



//TEST REMOVAL          query << "INSERT INTO articles_training (ar_title,
ar_body, ar_summary, ar_url, ar_date, cat_id) values
('"<<escape<<row["ar_title"].get_string() <<"',
'"<<escape<<row["ar_body"].get_string()<<"',
'"<<escape<<row["ar_summary"].get_string()<<"',
'"<<row["ar_url"].get_string()<<"', '"
<<row["ar_date"].get_string()<<"','"<<cat_id<<"' )";
//TEST REMOVAL     #ifdef SHOW_QUERIES
//TEST REMOVAL          clog<<"Query is:"<<query.preview()<<endl;
//TEST REMOVAL     #endif
//TEST REMOVAL          query.execute();
//TEST REMOVAL          query.reset();
//TEST REMOVAL          total_queries++;


                //find out what is the newly inserted article's id in the
articles_training table
                query << "select ar_id from mechanism_training.articles_training
where ar_title='"<<escape<<row["ar_title"].get_string()<<"'";
#ifdef SHOW_QUERIES
                clog<<"Query is:"<<query.preview()<<endl;
```

```
#endif
                mysqlpp::Result res3 = query.store();
                query.reset();
                total_queries++;
                int ar_id;
//              clog<<"Keywords size is: "<<res.size()<<endl;
                if (res3.size()==1) {//one article exists with this title this
is totally normal
                        mysqlpp::Row row;
                        row = res3.fetch_row();
                        std::istringstream ss(row["ar_id"].get_string());
                        ss >> ar_id;
                }
                else if (res3.size()>1){//more than one article exists, this
isn't so normal...
                        row = res3.fetch_row();
                        std::istringstream ss(row["ar_id"].get_string());
                        ss >> ar_id;

                }

                else{ //no articles exist with this title, this is absolutely
abnormal, EXIT NOW!!
                        clog<<"UNEXPECTED ERROR!!!"<<endl;
                        exit(-5);

                }









                /*Now add the text's data into the specified category*/
                //we need to insert the keywords first to the keywords_training
table*/

                for ( iter2 = mpsorted->begin(); iter2 != mpsorted->end();
iter2++ ) {
                //check to see if this keyword is already in the db
                        query << "select * from
mechanism_training.keywords_training where kw_name='"+iter2->second+"'";
#ifdef SHOW_QUERIES
                        clog<<"Query is:"<<query.preview()<<endl;
#endif
                        mysqlpp::Result res3 = query.store();
                        query.reset();
                        total_queries++;
                        int kw_id;
//              clog<<"Keywords size is: "<<res.size()<<endl;
                        if (res3.size()==1) {//keyword exists, update the
frequencies only
                        // Update frequencies for k/ws that already exist, to do
this we need to create a new record, or update the existing one in the
keyword2category table
                        //first find out the keyword's id
                                mysqlpp::Row row;
                                row = res3.fetch_row();
                                std::istringstream
ss(row["kw_id"].get_string());
                                ss >> kw_id;
//                      clog<<"Category id is:"<<cat_id<<endl;
//                      clog<<"Keyword id is:"<<kw_id<<endl;
                        //now check to see of the record kw_id-cat_id already
exists in the db
                                query << "select * from
mechanism_training.keywords_category_training where kw_id='"<<kw_id<<"' AND
cat_id='"<<cat_id<<"'";
#ifdef SHOW_QUERIES
                                clog<<"Query is:"<<query.preview()<<endl;
#endif
                                mysqlpp::Result res4 = query.store();
                                query.reset();
                                total_queries++;
                                if (res4.size()==1) { //record: kw_id-cat_id
already exists in the keyword2category table, update only frequencies
                                        query << "UPDATE
mechanism_training.keywords_category_training set
abs_frequency=abs_frequency+"<<iter2->first<<" where cat_id='"<<cat_id<<"' AND
kw_id='"<<kw_id<<"'";
```

```
#ifdef SHOW_QUERIES
                                        clog<<"Query
is:"<<query.preview()<<endl;
#endif
                                        //TEST REMOVAL    query.execute();
                                        clog<<"hehe.. i didn't execute this
query.."<<endl;
                                        query.reset();
                                        total_queries++;
                                }
                                else{ //record: kw_id-cat_id doesnt exist in the
keyword2category table, create it inserting also the frequencies
                                        query << "INSERT INTO
mechanism_training.keywords_category_training (cat_id, kw_id, abs_frequency)
values ('"<<cat_id<<"','"<<kw_id<<"','"<<iter2->first<<"')";
#ifdef SHOW_QUERIES
                                        clog<<"Query
is:"<<query.preview()<<endl;
#endif
                                        //TEST REMOVAL    query.execute();
                                        clog<<"hehe.. i didn't execute this
query.."<<endl;
                                        query.reset();
                                        total_queries++;


                                }

                        }
                        else   {//keyword doesn't exist, create it in the
keywords db first

                                query << "INSERT INTO
mechanism_training.keywords_training (kw_name) values ('"+iter2->second+"')";
#ifdef SHOW_QUERIES
                                clog<<"Query is:"<<query.preview()<<endl;
#endif
                                query.execute();
                                query.reset();
                                total_queries++;

                        //now find out whats the id of the newly created keyword
                                query << "select * from
mechanism_training.keywords_training where kw_name='"+iter2->second+"'";
#ifdef SHOW_QUERIES
                                clog<<"Query is:"<<query.preview()<<endl;
#endif
                                mysqlpp::Result res4 = query.store();
                                query.reset();
                                total_queries++;

                                mysqlpp::Row row;
                                row = res4.fetch_row();
                                std::istringstream
ss(row["kw_id"].get_string());
                                ss >> kw_id;
//                      clog<<"Category id is:"<<cat_id<<endl;
//                      clog<<"Keyword id is:"<<kw_id<<endl;

                        //now create a kw_id-cat_id record into keyword2category
table (since this keyword wasn't in the keywords table, no such record will
exist in keyword2category table, so create it
                                query << "INSERT INTO
mechanism_training.keywords_category_training (cat_id, kw_id, abs_frequency)
values ('"<<cat_id<<"','"<<kw_id<<"','"<<iter2->first<<"')";
#ifdef SHOW_QUERIES
                                clog<<"Query is:"<<query.preview()<<endl;
#endif
                                //TEST REMOVAL    query.execute();
                                clog<<"hehe.. i didn't execute this
query.."<<endl;
                                query.reset();
                                total_queries++;
                        }
                        //
```

```
                        //Take care of the keywords_articles_training table

                mysqlpp::Query query8 = con.query();
                query8 << "select kw_id from
mechanism_training.keywords_training WHERE kw_name='"<<iter2->second<<"'";
//Find out the kw_id
#ifdef SHOW_QUERIES
                clog<<"Query8 is:"<<query8.preview()<<endl;
#endif
                mysqlpp::Result res8 = query8.store();
                query8.reset();
                total_queries++;
                int kw_id2;

//      clog<<"Total records: "<<res.size()<<endl;
                if (res8.size()==1){ //pair exists, dont add it
                        mysqlpp::Row row8;
                        row8 = res8.fetch_row();
                        std::istringstream ss(row8["kw_id"].get_string());
                        ss >> kw_id2;
                        //insert keyword in the keywords_articles_training table
                        query8 << "INSERT INTO
mechanism_training.keywords_articles_training (ar_id, kw_id,frequency) values
('"<<ar_id<<"','"<<kw_id2<<"','"<<iter2->first<<"')";
#ifdef SHOW_QUERIES
                        clog<<"Query8 is:"<<query8.preview()<<endl;
#endif
                        query8.execute();
                        query8.reset();
                        total_queries++;
                }
                else{
                        clog<<"FATAL ERROR: KEYWORD DOESNT EXIST!!!"<<endl;
                        exit(-3);

                }









        }









                clog<<"Total queries passed to db: "<<total_queries<<endl;
                return t_art_ADDED_TO_TRAINING;
        }




        else{ //This text is not aimed to be a training set, just compare it
with the existing categories and store its relation with all available
categories

                mysqlpp::Query query = con.query();

                //First find all unique categories ids
                vector <int> category_ids;
                query << "select cat_id from category";
                clog<<"Query is:"<<query.preview()<<endl;
                mysqlpp::Result res1 = query.store();
```

```
                query.reset();

                if (res1.size()==0){
                        clog<<"No categories exist in the db, exiting..."<<endl;
                        exit(-3);
                }
                mysqlpp::Row row1;
                mysqlpp::Row::size_type i1;

                for (i1 = 0; row1 = res1.at(i1); ++i1) {
                        category_ids.push_back(row1.at(0));
                }

//              clog<<"Total categories found:"<<category_ids.size()<<endl;
//              for (int i=0;i<category_ids.size();i++){
//                      clog<<category_ids[i]<<endl;
//              }


                /** Categorization of the input text, according to the
categories already in the database starts from here **/

                //for each category registered in the database, find the
keywords that are related to this category
                map <string,double> cosine_simmilarities;
                for (int ii=0;ii<category_ids.size();ii++) {

                        string keywords_str;
                        for ( iter2 = mpsorted->begin(); iter2 !=
mpsorted->end(); iter2++ ) {
                                if (iter2!=mpsorted->begin()){
                                        keywords_str+=" OR ";
                                }
                                //keywords_str+="keywords.kw_name='";
                                keywords_str+="a.kw_name='";
                                keywords_str+=iter2->second;
                                keywords_str+="' ";


                        }
                        string category_name;
//                      query << "select
keywords.kw_id,keywords.kw_name,category.cat_id,category.cat_title,
keyword2category.abs_frequency from keywords,category,keyword2category where
(keywords.kw_id = keyword2category.keywords_kw_id) AND
(category.cat_id=keyword2category.category_cat_id) AND
(category.cat_id="<<category_ids[ii]<<") AND ("+ keywords_str+ ")";
//                      clog<<"Query is:"<<query.preview()<<endl;

                        query << "select
a.kw_id,a.kw_name,b.cat_id,b.cat_title,c.abs_frequency from keywords_training
a,category b, keywords_category_training c where (a.kw_id = c.kw_id) AND
(b.cat_id=c.cat_id) AND (b.cat_id="<<category_ids[ii]<<") AND ("+ keywords_str+
")";
                        clog<<"Query is:"<<query.preview()<<endl;

                        mysqlpp::Result res = query.store();
                        query.reset();
                        vector <Keyword_Frequency> category_keywords;
                        if (res.num_rows()>0) {
                        //if (res) {

                                mysqlpp::Row row;
                                mysqlpp::Row::size_type i;
                                Keyword_Frequency temp;
                                for (i = 0; row = res.at(i); ++i) {


temp.keyword=row["kw_name"].get_string();
                                        temp.abs_frequency=row.at(4);
                                        category_keywords.push_back(temp);

category_name=row["cat_title"].get_string();
                                //clog<<row.at(4)<<" "<<row.at(1)<<endl;
                                }
                        }
                        else {
                                //no rows returned from the db, continue the for
loop
                                continue;
```

```
                        }


                        clog<<"----------Printing data--------------"<<endl;
                        for (int i=0; i<category_keywords.size();i++){

clog<<i<<"Keyword:"<<category_keywords[i].keyword<<"
Frequency:"<<category_keywords[i].abs_frequency<<endl;
                        }                                                                   460


                        /*Now find the cosine relationship between the text's
keywords−frequency vector and the categorie's keywords-frequency vector*/

//                      clog<<"This is the text's keyword−frequency
vector"<<endl;
//                      for ( iter2 = mpsorted->begin(); iter2 !=
mpsorted->end(); iter2++ ) {
//                              clog<<iter2->second<<" "<<iter2->first<<endl;          470
//                      }

//                      clog<<"This is the categories' with id ==
"<<category_ids[ii]<<" keyword-frequency vector for the keywords that appear in
this text"<<endl;
//                      for (int i=0; i<category_keywords.size();i++){
//                              clog<<category_keywords[i].keyword<<"
"<<category_keywords[i].abs_frequency<<endl;
//                      }
                                                                                    480

                        /**Calculating the cosine metric between the 2
vectors**/
                        double dot_product=0;
                        double norm_text=0;
                        double norm_category=0;



                        double all_the_frequencies = 0;                             490
                        for ( iter2 = mpsorted->begin(); iter2 !=
mpsorted->end(); iter2++ ) {
                                all_the_frequencies += iter2->first;
                        }

                        for ( iter2 = mpsorted->begin(); iter2 !=
mpsorted->end(); iter2++ ) { //for each keyword in the text's keywords list

                                // prwta elegxoume an iparxei to keyword...
                                // an den iparxei to vazoume..                       500

                                        mysqlpp::Query query_vac =
con.query();
                                        query_vac << "select kw_id from
keywords WHERE kw_name='"<<iter2->second<<"'"; //Find out the kw_id
                                #ifdef SHOW_QUERIES
                                        clog<<"Query8
is:"<<query_vac.preview()<<endl;
                                #endif
                                        mysqlpp::Result res_vac_1 =                 510
query_vac.store();
                                        query_vac.reset();
                                        //total_queries++;
                                        int kw_id_vac;

                                //      clog<<"Total records:
"<<res.size()<<endl;
                                        if (res_vac_1.size()==0){ //no
keyword
                                                                                    520
                                                //insert keyword in the
keywords_articles_training table
                                                query_vac << "INSERT
INTO keywords (kw_id, kw_name) values ('','"<<iter2->second<<"')";
                                #ifdef SHOW_QUERIES
                                                clog<<"Query8
is:"<<query_vac.preview()<<endl;
                                #endif
                                                query_vac.execute();
                                                query_vac.reset();                  530
                                                query_vac << "select
kw_id from keywords WHERE kw_name='"<<iter2->second<<"'"; //Find out the kw_id
```

```
                                #ifdef SHOW_QUERIES
                                        clog<<"Query8
is:"<<query_vac.preview()<<endl;
                                #endif
                                        mysqlpp::Result res_vac_2 =
query_vac.store();
                                        query_vac.reset();
                                        mysqlpp::Row row_vac_2;                                 540
                                                row_vac_2 =
res_vac_2.fetch_row();
                                                std::istringstream
ss(row_vac_2["kw_id"].get_string());
                                                ss >> kw_id_vac;
                                                //total_queries++;
                                        }
                                        else{
                                                mysqlpp::Row row_vac;
                                                row_vac =                                       550
res_vac_1.fetch_row();
                                                std::istringstream
ss(row_vac["kw_id"].get_string());
                                                ss >> kw_id_vac;

                                        }
                                // edw exw parei to keyword id kai to exw valei
mesa sto keywords... twra tha prepei afou ipologizw gia kathe keyword to
rel_freq na to vazw sto keyword2artice...
                                        double the_real_rel_freq =                              560
((double)iter2->first)/all_the_frequencies;
                                        query_vac << "INSERT INTO
keyword2article (articles_ar_id, keywords_kw_id, frequency, rel_frequency,
positions) values ('"<<ar_id<<"','"<<kw_id_vac<<"','"<<iter2->first<<"','"<<
the_real_rel_freq<<"', '0')";
                                #ifdef SHOW_QUERIES
                                        clog<<"Query8
is:"<<query_vac.preview()<<endl;
                                #endif
                                        query_vac.execute();                                    570
                                        query_vac.reset();


                                for (int i=0; i<category_keywords.size();i++){
//for each keyword in this categorie's vector

//clog<<"iter2->second::"<<iter2->second<<"\t"<<"category_keywords[i].keyword::"
<<category_keywords[i].keyword<<endl;
                                        if
(iter2->second==category_keywords[i].keyword){                                                  580
                                                //clog<<"
dot_product="<<dot_product;

dot_product+=iter2->first*category_keywords[i].abs_frequency;
                                                //clog<<"
dot_product="<<iter2->first<<"*"<<category_keywords[i].abs_frequency<<"=="<<
dot_product;

                                                break;
                                        }                                                       590

                                }

                        }

                        //clog<<"dot_product::"<<dot_product<<endl;
                        //calculate the text's norm
                        for ( iter2 = mpsorted->begin(); iter2 !=
mpsorted->end(); iter2++ ) {
                                norm_text+=iter2->first*iter2->first;                           600

                        }
                        norm_text=sqrt(norm_text);
                        //clog<<"norm_text::"<<norm_text<<endl;
                        //calculate the categorie's norm
                        for (int i=0; i<category_keywords.size();i++){

norm_category+=category_keywords[i].abs_frequency*category_keywords[i].
abs_frequency;
                                                                                                610
                        }
                        norm_category=sqrt(norm_category);
                        //clog<<"norm_category::"<<norm_category<<endl;
```

```
//                        clog<<"Cosine is:
"<<dot_product/(norm_text*norm_category)<<" with category:
"<<category_name<<endl;
                        cosine_simmilarities.insert(make_pair( category_name,
(dot_product/(norm_text*norm_category)) ) );
                }

//              clog<<"Simmilarities of this text with the registered categories
are:"<<endl;
//              for( map<string,double>::iterator iter =
cosine_simmilarities.begin(); iter != cosine_simmilarities.end(); ++iter ) {
//                      clog <<iter->first<<" "<< iter->second << endl;
//              }


                /**compare_simmilarities and determine what we should do with
the article **/
                //DBL2STR cosine_simmilarities_sorted;



freq_sorted_keywords(cosine_simmilarities_sorted,&cosine_simmilarities);
                /*Relativities are now sorted ascending*/

                for( map<double,string>::iterator iter =
cosine_simmilarities_sorted->begin(); iter !=
cosine_simmilarities_sorted->end(); ++iter ) {
                        clog <<iter->first<<" "<< iter->second << endl;
                }
                map<double,string>::iterator iter =
cosine_simmilarities_sorted->end();
                double highest_sim = (--iter)->first;
                double sec_highest_sim = (--iter)->first;

                if (highest_sim >= (sec_highest_sim + 1.22)) { //edited 03082007
this was 0.22 and set to 1.22 so that no futher articles would be added to the
training set
                        /**If this article can be added to the dynamically
changing training set**/

                        return t_art_TRAINING;
                }

                else if (highest_sim >= (sec_highest_sim + 0.08)){
                        /**If we are able to categorize the article**/

                        return t_art_CATEGORIZED;
                }


                else {
                        /**If this article cannot be categorized**/

                        return t_art_UNCATEGORIZED;
                }



        }

}
```

Α.2 Πυρήνας κώδικα περίληψης

```
int mechanism::summarize(char * filename, int size, int kw_percentage,const char
* category, int round_characters, stringstream *summarySS, int user_id){

...
...
...


        /*SENTENCE RATING BEGINS HERE*/
        /*for each string in the vector ¡string¿ Sentences calculate its weight
*/
        double k1=1.4; //This is the extra weight we give if a keyword is also
in the title
        double k2=1.2; //This is the weight we give for the appearences of
keywords in the sentence
        int L=1; //this is the extra value added

        map <int,double> SentenceWeights;
//      for (int i=0;i<sentences->size();i++){
//              SentenceWeights.insert(make_pair( i,1.0) );//Add 1 in each
sentence's weight
        //
//      }
        srand(time(0));// Initialize random number generator.




        vector <Keyword_Frequency> keyword_frequency;

        if (user_id!=0) {
                /*a user_id is defined*/
                /*firstly acquire all the user keywords from the user_keyword
table*/
                clog<<"Connecting to database..."<<endl;
//      // Connect to the sample database.
                mysqlpp::Connection con(false);
                //
                extern char * host[4];//usage:[host] [user] [password] [port]



                if (!connect_to_db(5, host, con,"mechanism")) {
                        clog<<"Could not connect to database"<<endl;
                }
                mysqlpp::Query query = con.query();


                query << "SELECT b.kw_name, a.rel_frequency FROM user_keyword a
LEFT JOIN keywords_training b ON a.kw_id = b.kw_id WHERE user_id =
'"<<user_id<<"'";
#ifdef SHOW_QUERIES
                clog<<"Query is:"<<query.preview()<<endl;
#endif
                mysqlpp::Result res = query.store();
                query.reset();



                if (res) {

                        mysqlpp::Row row;
                        mysqlpp::Row::size_type i;
                        Keyword_Frequency temp;
                        for (i = 0; row = res.at(i); ++i) {
                                temp.keyword=row["kw_name"].get_string();
                                temp.rel_frequency=row.at(1);
                                keyword_frequency.push_back(temp);

                                //clog<< "Keyword is:"<<temp.keyword<<" with
rel_freq:"<<temp.rel_frequency<<endl;

                        }
                }

        }
```

```
        for (int i=0;i<Keywords_Sentences->size();i++){ //for each keyword i
                //clog<<"i:::"<<i<<endl;
                clog<<"Keyword:::: "<<Keywords_Sentences->at(i).keyword<<endl;

//              int test=rand()%2; //test is either 0 or 1
//              if (test==0)
//                      test=-1;
//              else
//                      test=1;
                //double k3 = ((rand() % ((18 +1) - 11)) + 11)*0.1*test;
//negative weight
                double k3,k4;


                /**Determine k3 factor**/
                if (strcmp(category,"none")==0) { //if no category is specified
by the command line
//                      k3 = ((rand() % ((18 +1) - 11)) + 11)*0.1;
//                      k4 = ((rand() % ((18 +1) - 11)) + 11)*0.1;
                        k3=1.0;
                        k4=1.0;

                        //do one of the above

                }
                else{
                        //a category is specified to the command line
parameters
                        //connect to the database to check the relation between
the keyword and this category

                        clog<<"Connecting to database..."<<endl;
//      // Connect to the sample database.
                        mysqlpp::Connection con(false);
                        //
                        extern char * host[4];//usage:[host] [user] [password]
[port]


                        if (!connect_to_db(5, host, con,"mechanism")) {
                                clog<<"Could not connect to database"<<endl;
                        }
                        mysqlpp::Query query = con.query();



                        query << "select
category.cat_title,keyword2category.abs_frequency from
keywords,category,keyword2category where keywords.kw_id =
keyword2category.keywords_kw_id AND category.cat_id =
keyword2category.category_cat_id AND keywords.kw_name = '"+
Keywords_Sentences->at(i).keyword+"'";
                        clog<<"Query is:"<<query.preview()<<endl;

                        mysqlpp::Result res = query.store();
                        query.reset();



                        vector <Category_Frequency> category_keywords;

                        if (res) {

                                mysqlpp::Row row;
                                mysqlpp::Row::size_type i;
                                Category_Frequency temp;
                                for (i = 0; row = res.at(i); ++i) {


temp.category=row["cat_title"].get_string();
                                        temp.abs_frequency=row.at(1);
                                        category_keywords.push_back(temp);


                                }
                        }

//                      clog<<"category_keywords is::"<<endl;
```

```
//                        for (int i=0;i<category_keywords.size();i++){
//                                clog<<category_keywords[i].category<<"
"<<category_keywords[i].abs_frequency<<endl;
                          //
//                        }

//                        int sum=0;
//                        int category_abs_freq=0;
//                        for (int i=0;i<category_keywords.size();i++){
//                                sum+=category_keywords[i].abs_frequency;
//                                if
(strcmp(category_keywords[i].category.c_str(),(const char *)category)==0){
//                                        //this is the record of the category
given as input
//                                        clog<<"YES"<<endl;
//
category_abs_freq=category_keywords[i].abs_frequency;
//                                }
                          //

//                        }
//                        if (category_abs_freq!=0){
//                                k3=(double)category_abs_freq/sum;
//                                clog<<"category_abs_freq
is::"<<category_abs_freq<<endl;
//                                clog<<"sum is::"<<sum<<endl;
//                                clog<<"k3::"<<k3<<endl;
//                                k3+=1;
                          //
//                        }
//                        else{
//                                k3=1;
//                        }

                          int sum=0;
                          int category_abs_freq=0;

                          for (int i=0;i<keyword_frequency.size();i++){

                                  if
(strcmp(keyword_frequency[i].keyword.c_str(),(const char *)category)==0){
                                          //this is the record of the category
given as input
                                          //clog<<"YES"<<endl;


category_abs_freq=category_keywords[i].abs_frequency;
                                  }



                          }
                          vector <double> a_i;

                          double c;
                          if (sum==0 && category_abs_freq==0)
                                  c=0;
                          else
                                  c=(double)category_abs_freq/sum;

                          for (int i=0;i<category_keywords.size();i++){

a_i.push_back((double)category_keywords[i].abs_frequency/sum);
                          }


                          //find out total categories number
                          query << "select count(*) from category";
#ifdef SHOW_QUERIES
                          clog<<"Query is:"<<query.preview()<<endl;
#endif
                          mysqlpp::Result res1 = query.store();
                          query.reset();
                          mysqlpp::Row row1;
                          row1 = res1.fetch_row();
                          std::istringstream ss(row1.at(0).get_string());
                          int total_categories;
                          ss >> total_categories;
```

```
                        double b=(double)1.0/total_categories;

//                      clog<< "b is::"<<b<<" c is::"<<c<<"category_abs_freq
is::"<<category_abs_freq<<"sum is::"<<sum<<endl;
                        if (c>b)
                                k3=1.0+(c-b);

                        else if (c<b)
                                k3=-1.0+(c-b);
                        else
                                k3=1.0;

                }



                /**Determine k4 factor**/
                int j;

                if (user_id!=0) {

                        for (j=0;j<keyword_frequency.size();j++){

                                if (keyword_frequency[j].keyword ==
Keywords_Sentences->at(i).keyword){
                                        //the text's keyword was found in the
keyword preferance list of the user
                                        if (keyword_frequency[j].rel_frequency >
0)

keyword_frequency[j].rel_frequency=keyword_frequency[j].rel_frequency+L;
                                        else if
(keyword_frequency[j].rel_frequency < 0)

keyword_frequency[j].rel_frequency=keyword_frequency[j].rel_frequency-L;
                                        k4 = keyword_frequency[j].rel_frequency
* Keywords_Sentences->at(i).total_occurences;
                                        break;
                                }



                        }

                        clog<<"Keyword:"<<Keywords_Sentences->at(i).keyword<<"
appears "<<Keywords_Sentences->at(i).total_occurences<<" times, K1=="<<k1<<"
K2=="<<k2<<" K3=="<<k3<<" K4=="<<k4<<" with user
preference:"<<keyword_frequency[j].rel_frequency<<endl;

                }



                clog<<"Keyword:"<<Keywords_Sentences->at(i).keyword<<" appears
"<<Keywords_Sentences->at(i).total_occurences<<" times, K1=="<<k1<<"
K2=="<<k2<<" K3=="<<k3<<" K4=="<<k4<<endl;



                if (k3 < 0 && k4 < 0) //special situation:: we dont want both
negatives because the result will be positive!!
                        k3 = -k3; //we inverse one of the k3, k4 factor and we
are done ;)

                for (int j=0;j<sentences->size();j++){ //for each sentence j
                        int
counter=funct1(Keywords_Sentences->at(i).sentences,Keywords_Sentences->at(i).
total_occurences,j);
                        //clog<<"Keyword::::
"<<Keywords_Sentences->at(i).keyword<<endl;
                        if (counter!=0){
                        //      clog<<"j:::"<<j<<endl;
                                if (Keywords_Sentences->at(i).sentences[0]==0){
//if this is a title keyword
```

```
SentenceWeights[j]+=(Keywords_Sentences->at(i).relative_frequency+L)*counter*(k1
+k2)*k3*k4;
                                        }
                                        else{//if this is not a title keyword


SentenceWeights[j]+=(Keywords_Sentences->at(i).relative_frequency+L)*counter*k2*
k3*k4;

                                                                                    330
                                        }
                                }

                        }



        }
//      clog<<"---------------------Sentence rating
results--------------------"<<endl;                                               340
//      for( map<int, double>::iterator iter = SentenceWeights.begin(); iter !=
SentenceWeights.end(); iter++ ) {
        //
//                      clog<<"["<<(*iter).first<<"]"<<(*iter).second<<endl;
        //
        //
//      }
        map <double,int> SentenceWeightsSorted;

        for ( map<int, double>::iterator iter = SentenceWeights.begin(); iter !=   350
SentenceWeights.end(); iter++ ){

                SentenceWeightsSorted.insert( map <double,int> ::value_type(
(*iter).second,(*iter).first ) );
        }




                                                                                    360
        vector <string> *tmp=lol.getRealSentences();
        clog<<"tmp size is:"<<tmp->size()<<endl;
        clog<<"tmp is:"<<endl;
        for (int i=0;i<tmp->size();i++) {
                clog<<i<<":"<<tmp->at(i)<<endl;

        }
        clog<<"sentences is:"<<endl;
        for (int i=0;i<sentences->size();i++) {
                clog<<i<<":"<<sentences->at(i)<<endl;                               370

        }

        clog<<"---------------------Sentence rating
sorted--------------------"<<endl;
        for( map<double, int>::iterator iter = SentenceWeightsSorted.begin();
iter != SentenceWeightsSorted.end(); iter++ ) {

                clog<<"["<<(*iter).second<<"]"<<(*iter).first<<" --->
"<<sentences->at((*iter).second)<<endl;                                             380
                clog<<tmp->at((*iter).second)<<endl; //print the sentence itself

        }

//      clog<<"Q. Give the percentage (%) of the sentences that you
want:"<<endl;
//      int answer=(int) sentence_percentage;
//      cin >> answer;
        int answer;
        int temp_flag=0;                                                            390
        for (int i=0;i<tmp->size();i++) {
                temp_flag+=tmp->at(i).size();
                if (temp_flag > round_characters){
                        clog << "temp_flag is::"<<temp_flag<<" round_characters
is:"<<round_characters<<endl;
                        clog << "tmp->at(i) is::"<<tmp->at(i)<<endl;
                        clog << "tmp->at(i).size()
is::"<<tmp->at(i).size()<<endl;
                        answer=i+1;
                        break;                                                      400
                }
```

```
        }
        if (temp_flag < round_characters){
                answer = tmp->size();
                clog<<"We will keep all of the "<<tmp->size()<<"
sentences"<<endl;
        }

        map<double, int>::reverse_iterator iter1 =
SentenceWeightsSorted.rbegin();

//      answer=(answer*SentenceWeightsSorted.size())/100;
        if (answer==0)
                answer=2;
        clog<<"Keeping "<<answer<<" of total "<<SentenceWeightsSorted.size()<<"
sentences"<<endl;
        vector <int> indexes;
        for(    int i=0;i<answer+1;i++) {
                indexes.push_back((*iter1).second);
                //clog<<tmp->at((*iter1).second)<<endl; //print the sentence
itself
                iter1++;
        }

//      clog<<"Indexes are:"<<endl;
//      for (int i=0;i<indexes.size();i++){
//              clog<<indexes[i]<<endl;
        //
//      }

        sort( indexes.begin(), indexes.end() );

        /*If the returned summary consists only of title, we send the title*/
        if (answer==1) {
                *summarySS<<tmp->at(indexes[0])<<". "<<endl;
                clog << "Kept a total of "<<answer <<" sentences title not
removed from the summary"<<endl;
        }
        else {
        /*else we dont use the title in the returned summary*/
        for(    int i=1;i<answer;i++) {
                //cout<<"["<<i<<"]"<<tmp->at(indexes[i])<<". "<<endl; //print
the sentence itself
                //cout<<"i::"<<i<<endl;
                *summarySS<<tmp->at(indexes[i])<<". "<<endl;

        }
        clog << "Kept a total of "<<answer-1 <<" sentences having removed the
title"<<endl;
        }
```

Α.3 Ροή διεργασιών

```
/**
Usage: pers_sum [OPTION...]
summarization - categorization - personalization mechanism

  -c, --categorize              Categorize
  -d, --db                      Use database connection
  -h, --db_host                 MySQL Database connection host
  -i, --input=FILE              Input from XML file instead
  -o, --output=FILE             Output to FILE instead of standard output
  -p, --db_pw                   MySQL Database connection password
  -P, --kw_perc                 Percentage [0-100%] of keywords to keep from
                                summarization or categorization process
  -q, --quiet                   Don't produce any output
  -R, --noRSS                   Don't produce summary of articles in RSS format
                                (RSS format is the default)
  -s, --summarize               Summarize
  -S, --sent_perc               Percentage [0-100%] of sentences to keep for the
                                summarization procedure
  -t, --training                Parse existing articles of the articles_training
table in order to k/w extract them
  -u, --username=NAME           Username used for personalized summary
  -U, --db_user                 MySQLDatabase connection username
  -v, --verbose                 Produce verbose output
  -?, --help                    Give this help list
      --usage                   Give a short usage message
  -V, --version                 Print program version

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

**/
...
...
...


/** This is used to parse the articles residing in the
mechanism_training.articles_training table that have flag==0 and k/w extract
them. We populate the tables::      */
        if (arguments.training == 1) {
                clog << "Running as training set"<<endl;
                char * in_file=NULL;
                char * category="none";



                /**Check the 'mechanism_training.articles_training' table to
find any articles with flag==0 (unparsed) and attempt a training base
population**/

                clog<<"Connecting to database..."<<endl;
                mysqlpp::Connection con(false);

                if (!connect_to_db(5, host, con,"mechanism")) {
                        return -2;
}
                mysqlpp::Query query = con.query();
                query << "SELECT a.ar_id, a.ar_title, a.ar_body, b.cat_title
FROM mechanism_training.articles_training a JOIN mechanism_training.category b
ON a.cat_id=b.cat_id where a.flag = 0";
                //clog<<"Query is:"<<query.preview()<<endl;

                string ar_id="";
                string ar_title="";
                string body="";
                string type="";

                mysqlpp::Result res = query.store();

                if (res.empty())
                        clog<<"No articles for addition to the training set were
found"<<endl;
                query.reset();

                mysqlpp::Row row;
                stringstream summarySS;
```

```
                while(row = res.fetch_row()){
                        summarySS.str(""); //clear the stringstream
                        ar_id=row["ar_id"].get_string();
                        std::istringstream ss(ar_id);
                        int ar_id_int;
                        ss >> ar_id_int;

                        ar_title=row["ar_title"].get_string();
                        if (ar_title=="")
                                continue; //leave this article, continue with
the next
                        body=row["ar_body"].get_string();
                        if (body=="")
                                continue; //leave this article, continue with
the next
                        type=row["cat_title"].get_string();
                        if (type=="")
                                continue; //leave this article, continue with
the next

                        string tmp1="train/"+ar_id;
                        ofstream myfile (tmp1.c_str());


                        clog<<"<text><id>"<<ar_id<<"</id><type>"<< type
<<"</type><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<endl;
                        myfile<<"<text><id>"<<ar_id<<"</id><type>"<< type
<<"</type><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<endl;
                        myfile.flush();
                        myfile.close();

                        char filenameee [20]="";
                        strcpy(filenameee,tmp1.c_str());

                        if
((mech3.categorize(filenameee,size,1,kw_percentage,0,ar_id_int,NULL))!=
t_art_ADDED_TO_TRAINING) {
                                clog<<"Could not add article
(article_id=="<<ar_id_int<<") to training set"<<endl;
                                exit(-1);
}
                        update_training_flag (ar_id_int, 1); //articles_training
flag is set to 1 when the text has been parted (PHP code handles the rest...)
                        //break; /**REMOVE ME*/


}
}


        if (arguments.categorize==1) {
                /*Categorize a given article*/


                char * in_file=NULL;
                char * category="none";



                /**Check the 'articles' table to find any articles with
categorized flag==0 (uncategorized) and attempt a categorization**/

                clog<<"Connecting to database..."<<endl;
                mysqlpp::Connection con(false);

                if (!connect_to_db(5, host, con,"mechanism")) {
                        return -2;
}
                mysqlpp::Query query = con.query();
                query << "select ar_id, ar_title, ar_body from articles where
flag_categorized = 0";
                //clog<<"Query is:"<<query.preview()<<endl;

                string ar_id="";
                string ar_title="";
                string body="";

                mysqlpp::Result res = query.store();

                if (res.empty())
```

```
                clog<<"No uncategorized articles were found"<<endl;
        query.reset();

        mysqlpp::Row row;
        stringstream summarySS;

        int counter=0;

        while(row = res.fetch_row()){
                summarySS.str(""); //clear the stringstream
                ar_id=row["ar_id"].get_string();
                std::istringstream ss(ar_id);
                int ar_id_int;
                ss >> ar_id_int;

                ar_title=row["ar_title"].get_string();
                if (ar_title=="")
                        continue; //leave this article, continue with
the next
                body=row["ar_body"].get_string();
                if (body=="")
                        continue; //leave this article, continue with
the next
                string tmp1="cat/"+ar_id;
                ofstream myfile (tmp1.c_str());

myfile<<"<text><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>";

clog<<"<text><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<endl;
                myfile.flush();

                myfile.close();
                char filenameee [20]="";
                strcpy(filenameee,tmp1.c_str());


                //error_codes art_status = t_art_STATUS_UNKNOWN;
//initialization
                map<double,string> cosine_simmilarities_sorted;
                map<double,string> cosine_simmilarities_sorted_summary;
                t_art_STATUS art_status =
mech1.categorize(filenameee,size,0,kw_percentage,0,ar_id_int,&
cosine_simmilarities_sorted);
                if (art_status == t_art_UNCATEGORIZED){ /*If we were not
able to categorize the article, procceed with default summarization*/
                        clog<<"This article can't be categorized
:("<<endl;
                        clog<<"Trying to categorize
summarization"<<endl;

mech2.summarize_perc(filenameee,size,kw_percentage,"none",sentence_percentage,&
summarySS,0);
                        /*Try to categorize the summary*/
                        clog<<"Summary is: "<<  summarySS.str()<<endl;
                        myfile.open(tmp1.c_str());

myfile<<"<text><title>"<<ar_title<<"</title><body>"<<summarySS.str()<<"</body></
text>";
                        myfile.flush();
                        myfile.close();
                        if ((art_status =
mech1.categorize(filenameee,size,0,100,0,ar_id_int,&
cosine_simmilarities_sorted_summary))==t_art_UNCATEGORIZED){
                                clog<<"Summary could not be categorized
:("<<endl;
                                clog<<"Saving relativities of the whole
text with the categories"<<endl;
                                if
(insert_article2category_relativities(ar_id_int,&cosine_simmilarities_sorted,1)!
=t_art_STATUS_OK) {
                                        clog<<"DATABASE ERROR at saving
relativities of the whole text (article_id=="<<ar_id_int<<") with the
categories"<<endl;
                                        exit (-1);
}

}
                        else{
                                /**Summary cound be categorized**/
                                clog<<"Summary could be categorized
;)"<<endl;
```

```
                                        clog<<"Saving relativities of the text's summary with the categories"<<endl;
                                        if (insert_article2category_relativities(ar_id_int,&cosine_simmilarities_sorted_summary,2)!=t_art_STATUS_OK) {
                                                clog<<"DATABASE ERROR at saving relativities of the of the text's summary (article_id=="<<ar_id_int<<") with the categories"<<endl;
                                                exit (-1);
                                        }

                                }

                        }

                        else if (art_status == t_art_TRAINING) {
                                clog<<"This article can be added to training set :)"<<endl;
                                /*Fix the XML input file to have this format:: <text><type>(category)</type><title>...</title><body>...</body>*/
                                myfile.open(tmp1.c_str());

                                map<double,string>::iterator iter = cosine_simmilarities_sorted.end();
                                iter--;
                                //string highest_cat = iter->second;

                                clog<<"<text><id></id><type>"<< iter->second <<"</type><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<endl;
                                myfile<<"<text><id></id><type>"<< iter->second <<"</type><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>";
                                myfile.flush();
                                myfile.close();
                                if ((mech3.categorize(filenameee,size,1,kw_percentage,0,ar_id_int,&cosine_simmilarities_sorted))!=t_art_ADDED_TO_TRAINING) {
                                        clog<<"Could not add article (article_id=="<<ar_id_int<<") to training set"<<endl;
                                        exit(-1);
                                }
                                update_categorization_flag (ar_id_int, 3);
//categorization flag is set to 3 when the text was added to the training set




                                /**proceed with default summary**/

mech2.summarize_perc(filenameee,size,kw_percentage,"none",sentence_percentage,&summarySS,0);
}

                        else if (art_status == t_art_CATEGORIZED) {
                                clog<<"This article could be categorized ;)"<<endl;
                                clog<<"Inserting relativities in the db"<<endl;
                                /** Insert the simmilarities into the database and mark the article as 'categorized' **/
                                if (insert_article2category_relativities(ar_id_int,&cosine_simmilarities_sorted,1)!=t_art_STATUS_OK) {
                                        clog<<"DATABASE ERROR at saving relativities of the whole text (article_id=="<<ar_id_int<<") with the categories"<<endl;
                                        exit (-1);
                                }

                                /**proceed with default summary**/

mech2.summarize_perc(filenameee,size,kw_percentage,"none",sentence_percentage,&summarySS,0);
}

                        else {
                                clog<<"--------UNKNOWN ERROR--------"<<endl;
                                return -1;
```

```
}
                        clog<<"Inserting default (uncategorized) summary"<<endl;
                        if
(insert_articleSummary(ar_id_int,&summarySS)!=t_art_STATUS_OK) {
                                clog<<"DATABASE ERROR at inserting summary
(article_id=="<<ar_id_int<<")"<<endl;
                                exit (-1);
}
                        counter++;                                                      330
}


        clog<<"Updated "<<counter<<" records"<<endl;









                                                                                        340


}
        /**We want RSS feed of the summarized articles**/
        if (arguments.Rss==1) {
                try {                                                                   350


                        cgicc::Cgicc formData;
                        rssfeed << cgicc::HTTPContentHeader("text/xml");
                        rssfeed << "<?xml version=\"1.0\"
encoding=\"ISO-8859-1\"?>"<<endl;
                        //cout <<
cgicc::HTMLDoctype(cgicc::HTMLDoctype::eStrict) << endl;
                        //rssfeed<<"<?xml version=\"1.0\"?>";
                        const cgicc::CgiEnvironment& env =                              360
formData.getEnvironment();
                        string remoteuser="--";
//                      remoteuser = env.getRemoteUser();
                        cgicc::form_iterator fvalue1;
                        int defined_username=0;
                        fvalue1 = formData.getElement("name");

                        if( fvalue1 != (*formData).end() && !fvalue1->isEmpty()
) {
                                clog << "name: " << **fvalue1 << endl;                   370
                                remoteuser=**fvalue1;
                                defined_username=1;

}
                        else {
                                clog << "No text entered for name" << endl;
                                defined_username=0;


                                                                                        380
}




                        clog<<"remoteuser is:"<<remoteuser<<endl;
                        clog<<"defined_username is:"<<defined_username<<endl;
                        if (defined_username==1) {//We are using CGI web
interface
                                //arguments.username=(char *)remoteuser.c_str();        390

clog<<"strcpy(arguments.username,remoteuser.c_str())"<<endl;
                                char temp_name [40];
                                strcpy(temp_name,remoteuser.c_str());
                                arguments.username=temp_name;
                                clog<<"completed!"<<endl;
//                      clog<<"remoteuser is now:"<<remoteuser<<endl;
//                      string serversw = env.getServerSoftware();
//                      string clientsw = env.getUserAgent();
//                      string authtype = env.getAuthType();                            400
```

```
                //cout<<"Content-type: text/html\n\n";


//                          cgicc::Cgicc formData;
//                          cout << cgicc::HTTPHTMLHeader() << endl;
//                          cout << cgicc::HTMLDoctype(cgicc::HTMLDoctype::eStrict)
<< endl;
//      //cout<<"Content-type: text/html\n\n";                                    410
}
                          rssfeed<<"<rss version=\"2.0\">\n";

                          rssfeed<<"<channel><title>peRSSonal, News RSS
Summarizer</title>\n";

                          rssfeed<<"<description>"<<"This is the News Summarizer
mechanism presentation</description>\n";

rssfeed<<"<link>http://150.140.141.40:8081/rss</link>\n";                         420

                          rssfeed<<"<language>el</language>\n";

rssfeed<<"<docs>http://ru6.cti.gr</docs><pubDate>"<<time2str()<<"</pubDate>\n";

                          /**Resume adding one by one each rss feed**/


}
                catch(exception& e) {                                             430
        // handle any errors here.
                          clog <<"Exception: "<< e.what() << endl;
                          exit(-1);
}

                                                                                  440

}

                                                                                  450

        if (arguments.summarize==1) {
                clog<<"arguments.username is::"<<arguments.username<<endl;
                /*Categorize a given article*/

                char * in_file=NULL;                                              460
                char * category="none";


                if (strcmp(arguments.username,"-")==0){ //no username specified,
proceed with default summary
                          clog<<"no username is specified"<<endl;
                          clog<<"producing default summaries"<<endl;

                          clog<<"Connecting to database..."<<endl;                470
                          mysqlpp::Connection con(false);

                          if (!connect_to_db(5, host, con,"mechanism")) {
                                  return -2;
}
                          mysqlpp::Query query = con.query();
                          query << "SELECT ar_id, ar_title, ar_summary, ar_url,
ar_date FROM articles ORDER BY ar_date DESC LIMIT 0 , 15";
                          clog<<"Query is:"<<query.preview()<<endl;
                          mysqlpp::Result res = query.store();                    480
                          query.reset();
                          string ar_id="";
```

```
                        int ar_id_int=-1;
                        string ar_title="";
                        string ar_summary="";
                        string ar_url="";
                        string ar_date="";


                                                                                                    490

                        mysqlpp::Row row;
                        if (res.empty()){
                                clog<<"No articles found in the 'articles'
db"<<endl;
                                return -1;
}
                        stringstream summarySS;
                        while(row = res.fetch_row()){
                                summarySS.str(""); //clear the stringstream                         500
                                ar_id=row["ar_id"].get_string();
                                std::istringstream ss(ar_id);
                                int ar_id_int;
                                ss >> ar_id_int;
                                ar_title=row["ar_title"].get_string();
                                ar_summary=row["ar_summary"].get_string();
                                ar_url=row["ar_url"].get_string();
                                ar_date=row["ar_date"].get_string();

//                              string tmp1="sum/"+ar_id;                                           510
//                              ofstream myfile (tmp1.c_str());
//
myfile<<"<text><title>"<<ar_title<<"</title><body>"<<ar_body<<"</body></text>";
//
//clog<<"<text><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<
endl;
//                              myfile.flush();
//
//                              myfile.close();
//                              char filenameee [20]="";                                            520
//                              strcpy(filenameee,tmp1.c_str());
//
//
mech1.summarize(filenameee,size,kw_percentage,"none",sentence_percentage,&
summarySS,0);
                                if (arguments.Rss==1) {//produce summary of
articles
                                        rssfeed<<"      <item>\n";
                                        rssfeed<<"
<title>"<<ar_title<<"</title>\n";                                                                   530
                                        rssfeed<<"<link>"<<ar_url<<"</link>\n";

rssfeed<<"<pubDate>"<<ar_date<<"</pubDate>\n";

rssfeed<<"<description>"<<ar_summary<<"</description>\n";
                                        rssfeed<<"      </item>\n";




                                                                                                    540


}
                                else {
                                        clog<<"SUMMARY
IS:::::"<<ar_summary<<endl;

}
                                                                                                    550
}
                        if (arguments.Rss==1) { //send default summary
                                rssfeed<<"</channel>\n";
                                rssfeed<<"      </rss>\n";
                                /**Finally, send the rss response to the user**/
                                clog<<"Sending the default rss response to the
user..."<<endl;
                                cout<<rssfeed.str();

}                                                                                                   560
```

```
}
                else { //username is specified, produce personalized summary
                        /**Find out if this user exists in the database and find
his user_id**/
                        clog<<"username is specified"<<endl;
                        clog<<"Connecting to database..."<<endl;
                        mysqlpp::Connection con(false);


                        if (!connect_to_db(5, host, con,"mechanism")) {
                                return -2;
}
                        mysqlpp::Query query = con.query();
                        //query << "SELECT id, username, scr_width, scr_height
FROM user_rss WHERE username = '"<<arguments.username<<"'";
                        query << "SELECT id, username, scr_width, scr_height
FROM user_rss WHERE username = '"<<arguments.username<<"'";
                        clog<<"Query is:"<<query.preview()<<endl;

                        string id="";
                        string username_="";
                        string scr_width="";
                        string scr_height="";


                        mysqlpp::Result res = query.store();
                        query.reset();
                        if (res.empty()){
                                clog<<"No user found with this
name:"<<arguments.username<<endl;
                                if (arguments.Rss==1) { /*inform the rss user to
register to the service first*/
                                        rssfeed<<"       <item>\n";
                                        rssfeed<<"       <title>Please Register
to PeRSSonal system first</title>\n";

rssfeed<<"<link>http://150.140.141.40:8081/rss</link>\n";
                                        rssfeed<<"<pubDate></pubDate>\n";
                                        rssfeed<<"<description>Please Register
to peRSSonal system first before using the service. You can always receive a
default RSS Feed containing default (uncategorized and unpersonalized) summaries
of articles</description>\n";
                                        rssfeed<<"       </item>\n";
                                        rssfeed<<"</channel>\n";
                                        rssfeed<<"       </rss>\n";
                                        clog<<"Sending the rss response to the
user..."<<endl;
                                        cout<<rssfeed.str();

}
                                return -1;
}
                        else if (res.size()>1) {
                                clog<<"PROBLEM WITH THE DATABASE:: more than one
user found with the username::"<<arguments.username<<endl;
                                return -1;
}


                        mysqlpp::Row row;


                        int counter=0;

                        row = res.fetch_row();

                        id=row["id"].get_string();
                        username_=row["username"].get_string();
                        scr_width=row["scr_width"].get_string();
                        scr_height=row["scr_height"].get_string();
                        std::istringstream ss(id);
                        int user_id;
                        ss >> user_id;

                        query << "SELECT characters FROM resolution_chars WHERE
scr_width = '"<<scr_width<<"' AND scr_height = '"<<scr_height<<"'";
```

```
                        clog<<"Query is:"<<query.preview()<<endl;
                        mysqlpp::Result res1 = query.store();
                        query.reset();
                        int total_chars;
                        if (res1.empty()){
                                clog<<"No registered resolution with
width:"<<scr_width<<" and height:"<<scr_height<<endl;
                                clog<<"Setting total_chars of all summaries to
100"<<endl;
                                total_chars = 1000;
}
                        else {

                                mysqlpp::Row row1;

                                string total_chars_str="";

                                row1 = res1.fetch_row();

                                total_chars_str=row1["characters"].get_string();
                                std::istringstream ss_chars(total_chars_str);

                                ss_chars >> total_chars;


}


                        //query << "SELECT ar_id, ar_title, ar_body, ar_url,
ar_date FROM articles ORDER BY ar_date DESC LIMIT 0 , 10";
//                      query << "SELECT DISTINCT a.ar_id, a.ar_title,
a.ar_body, a.ar_url, a.ar_date FROM user_category_pref b LEFT JOIN
article2category c ON b.cat_id = c.cat_id LEFT JOIN articles a ON a.ar_id =
c.ar_id WHERE b.preference > 2 AND b.user_id = user_id ORDER BY a.ar_date DESC
LIMIT 0 , 10";


                        query << "SELECT DISTINCT a.ar_id, a.ar_title,
a.ar_body, a.ar_url, a.ar_date FROM user_category_pref b JOIN article2category c
ON b.cat_id = c.cat_id AND c.frequency >='"<<MIN_CAT_FREQUENCY<<"' AND
b.preference >='"<<MIN_PREFERANCE<<"' JOIN articles a ON a.ar_id = c.ar_id WHERE
b.user_id = user_id ORDER BY b.preference, a.ar_date DESC LIMIT 0 , 15";

                        clog<<"Query is:"<<query.preview()<<endl;

                        string ar_id="";
                        int ar_id_int=-1;
                        string ar_title="";
                        string ar_body="";
                        string ar_url="";
                        string ar_date="";

                        res = query.store();
                        query.reset();
                        if (res.empty()){
                                clog<<"No articles found in the 'articles'
db"<<endl;
                                return -1;
}
                        //total_chars=total_chars/res.size();
                        if (total_chars<40) {
                                clog<<"total_chars is::"<<total_chars<<",
setting to 40"<<endl;
                                total_chars=40;

}
                        stringstream summarySS;
                        while(row = res.fetch_row()){
                                summarySS.str(""); //clear the stringstream
                                ar_id=row["ar_id"].get_string();
                                std::istringstream ss(ar_id);
                                int ar_id_int;
                                ss >> ar_id_int;
                                ar_title=row["ar_title"].get_string();
                                ar_body=row["ar_body"].get_string();
                                ar_url=row["ar_url"].get_string();
                                ar_date=row["ar_date"].get_string();
```

```
                                        string tmp1="sum/"+ar_id;
                                        ofstream myfile (tmp1.c_str());

myfile<<"<text><title>"<<ar_title<<"</title><body>"<<ar_body<<"</body></text>";
                                                                                    730
//clog<<"<text><title>"<<ar_title<<"</title><body>"<<body<<"</body></text>"<<
endl;
                                        myfile.flush();

                                        myfile.close();
                                        char filenameee [20]="";
                                        strcpy(filenameee,tmp1.c_str());


mech1.summarize(filenameee,size,kw_percentage,"none",total_chars,&summarySS,                740
user_id);

                                        if (arguments.Rss==1) {//produce summary of
articles
                                                rssfeed<<"        <item>\n";
                                                rssfeed<<"
<title>"<<ar_title<<"</title>\n";
                                                rssfeed<<"<link>"<<ar_url<<"</link>\n";

rssfeed<<"<pubDate>"<<ar_date<<"</pubDate>\n";                                              750

rssfeed<<"<description>"<<summarySS.str()<<"</description>\n";
                                                rssfeed<<"        </item>\n";









                                                                                    760
}
                                        else {
                                        clog<<"SUMMARY IS:::::"<<summarySS.str()<<endl;

}

}


                                if (arguments.Rss==1) { //produce the closing of the RSS    770
feed
                                        rssfeed<<"</channel>\n";
                                        rssfeed<<"        </rss>\n";
                                        /**Finally, send the rss response to the user**/
                                        clog<<"Sending the rss response to the user
"<<username_<<endl;
                                        cout<<rssfeed.str();
}
                                                                                    780









}

}

                                                                                    790
        exit (0);
}
```