Ατομική Διπλωματική Εργασία

ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ ΣΥΣΤΑΣΕΩΝ ΓΙΑ ΚΡΙΤΕΣ ΕΠΙΣΤΗΜΟΝΙΚΩΝ ΑΡΘΡΩΝ

Χρίστος Ελευθερίου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Υλοποίηση Συστήματος Συστάσεων για Κριτές Επιστημονικών Άρθρων

Χρίστος Ελευθερίου

Επιβλέπουσα Καθηγήτρια
Άννα Φιλίππου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2023

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια της Διπλωματικής μου Εργασίας Κ. Άννα Φιλίππου, για την καθοδήγηση και την υποστήριξη που μου παρείχε καθ΄ όλη τη διάρκεια της διατριβής μου. Οι προτάσεις, οι υποδείξεις, τα σχόλια, αλλά και το θετικό κλίμα στη συνεργασία μας συνέβαλαν καθοριστικά στη διαμόρφωση της εργασίας μου και με βοήθησαν να πετύχω το καλύτερο δυνατό αποτέλεσμα.

Θα ήθελα επίσης να ευχαριστήσω την οικογένεια μου για την κατανόηση, την υποστήριξη και την ενθάρρυνση που μου έδωσαν κατά τη διάρκεια αυτής της προσπάθειας, αλλά και γενικότερα κατά την διάρκεια των σπουδών μου.

Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου που με ενθάρρυναν να δουλέψω, και για την θετική ενέργεια που μου μετέδιδαν κάθε μέρα.

Περίληψη

Η Ατομική Διπλωματική μου Εργασία έχει ως σκοπό την υλοποίηση ενός συστήματος συστάσεων (recommender system) για κριτές επιστημονικών άρθρων.

Η ιδέα αυτή προέκυψε μετά από την παρατήρηση πως η διαδικασία επιλογής κριτών για επιστημονικά άρθρα θα μπορούσε να αυτοματοποιηθεί, γλιτώνοντας χρόνο και κόπο στους μέχρι τώρα υπεύθυνους της διαδικασίας αυτής.

Οι στόχοι που είχαν τεθεί από την αρχή για την εργασία μου ήταν: με την είσοδο των ονομάτων κάποιων συγγραφέων και κάποιων λέξεων-κλειδιών, να εμφανίζονται άλλοι συγγραφείς οι οποίοι θα είναι οι πιο κατάλληλοι για κριτές σε άρθρο των εν λόγω συγγραφέων, με την χρήση πληροφοριών που βρίσκονται αξιόπιστη διαθέσιμη βάση δεδομένων. Ως στόχος είχε επίσης τεθεί τα αποτελέσματα που θα παρουσιάζονται στον χρήστη, να επεξηγούνται κάπως σε αυτόν με την παρουσίαση παραπάνω πληροφορίων έτσι ώστε να κατανοεί γιατί προέκυψαν τα αποτελέσματα που έχει μπροστά του.

Αποτέλεσμα της Ατομικής Διπλωματικής μου Εργασίας είναι η υλοποίηση ενός συστήματος κάνει την πιο πάνω διαδικασία. Συγκεκριμένα, το σύστημα χρησιμοποιεί την βάση δεδομένων του DBLP για πρόσβαση στα ονόματα των διαφόρων συγγραφέων αλλά και στα άρθρα που έχουν γράψει. Χρησιμοποιώντας τον αλγόριθμο machine learning *k*-Nearest-Neighbors αλλά και αλγόριθμο για λέξεις-κλειδιά, το σύστημα επιστρέφει αποτελέσματα που είναι αξιόπιστα και τηρούν τις προδιαγραφές που πρέπει.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Γενική Εισαγωγή

Η ιδέα της Διπλωματικής μου Εργασίας προτάθηκε από την Επιβλέπουσα Καθηγήτρια της Ατομικής Διπλωματικής μου Εργασίας, και είχε ως στόχο την ανάπτυξη ενός συστήματος συστάσεων (recommender system) με το οποίο θα γινόταν αυτόματα η διαδικασία επιλογής κριτών για επιστημονικά άρθρα. Ένα σύστημα, δηλαδή, το οποίο όταν πάρει ως είσοδο τους συγγραφείς ενός επιστημονικού άρθρου, και κάποιες λέξεις κλειδιά που να περιγράφουν το εν λόγω άρθρο, να προτείνει κριτές για αυτό, δηλαδή ερευνητές που να είναι ειδικοί στο θέμα που πραγματεύεται το άρθρο, αξιοποιώντας τα κοινά συνέδρια ή άρθρα μεταξύ των συγγραφέων και των πιθανών κριτών, αλλά και τις λέξεις κλειδιά που δόθηκαν.

1.2 Στόχος

Στόχος της Ατομικής Διπλωματικής μου Εργασίας ήταν η ανάπτυξη ενός συστήματος συστάσεων (recommender system) για κριτές επιστημονικών άρθρων. Το σύστημα αυτό θα πρέπει να είναι εύκολο στη χρήση για τον χρήστη, και να παρέχει ορθά και αξιόπιστα αποτελέσματα, και η λειτουργία του θα βασίζεται στα προϋπάρχοντα δεδομένα στην βάση δεδομένων της ιστοσελίδας DBLP, σε αλγόριθμους μηχανικής μάθησης (machine learning), αλλά και σε αλγόριθμο που θα λαμβάνει υπόψη τις λέξεις κλειδιά.

1.3 Περιγραφή και Συνεισφορά Εργασίας

Το σύστημα θα είναι προσβάσιμο μέσω web browsers όπως Google Chrome, Mozilla Firefox κλπ., που υποστηρίζονται από όλες τις σύγχρονες ηλεκτρονικές συσκευές. Στην σελίδα της εφαρμογής υπάρχουν τρία πεδία εισόδου στα οποία οι χρήστες θα πρέπει να δώσουν τις επιθυμητές για αυτούς εισόδους. Συγκεκριμένα, στο πρώτο πεδίο θα πρέπει να δώσουν όλα τα ονόματα των συγγραφέων που τους ενδιαφέρει, στο δεύτερο πεδίο θα πρέπει να δώσουν τις λέξεις-κλειδιά που θέλουν, και στο τρίτο πεδίο εισόδου το πόσα αποτελέσματα θέλουν να εμφανιστούν στο τέλος. Αφού οι χρήστες δώσουν τις επιθυμητές εισόδους και πατήσουν το κουμπί που καθορίζει το αν οι συστάσεις θα βασιστούν σε κοινά συνέδρια ή σε κοινά άρθρα, ξεκινά η διαδικασία εύρεσης κατάλληλων κριτών.

Μέσω της JavaScript περνιούνται τα στοιχεία σε πρόγραμμα γραμμένο σε PHP που βρίσκεται στην πλευρά του διακομιστή, και μέσω αυτού εκτελείται script γραμμένο σε Python με τις κατάλληλες εισόδους, το οποίο εφαρμόζει τους αλγόριθμους που χρειάζονται.

Χρησιμοποιώντας δεδομένα που βρίσκονται στη βάση δεδομένων της ιστοσελίδας DBLP[1], αλλά και τις εισόδους του χρήστη, εφαρμόζεται ο classification αλγόριθμος μηχανικής μάθησης *k* Nearest Neighbors[2] για να βρεθούν οι *k* πιο ‘κοντινοί’ συγγραφείς με αυτούς που έδωσε ως είσοδο ο χρήστης, με βάση τα κοινά μεταξύ τους συνέδρια ή άρθρα, με βάση το κουμπί που πάτησε ο χρήστης.

Στην συνέχεια, πάνω στα αποτελέσματα που προκύπτουν από τον αλγόριθμο *k* Nearest Neighbors, αποκλείονται όλοι οι συν-συγγραφείς με τους συγγραφείς εισόδου, και εφαρμόζεται ο αλγόριθμος που λαμβάνει υπόψη τις λέξεις-κλειδιά που έδωσε ο χρήστης. Συγκεκριμένα, για κάθε έναν από τους *k* συγγραφείς, μετρούνται οι εμφανίσεις των εν λόγω λέξεων-κλειδιών στους τίτλους των άρθρων που έχουν γράψει, και στο τέλος ταξινομούνται με αυτό τον αριθμό και κρατούνται οι 100 ‘κορυφαίοι’.

Τα αποτελέσματα αυτά παρουσιάζονται στη σελίδα του συστήματος σε μορφή ενός πίνακα που δείχνει το όνομα του συγγραφέα, το ‘score’ που έχει για τις λέξεις-κλειδιά με τη δυνατότητα προβολής των τίτλων των άρθρων τους που περιέχουν την κάθε λέξη-κλειδί με το πάτημα του αντίστοιχου κουμπιού, τα κοινά συνέδρια/περιοδικά με τους συγγραφείς εισόδου, και το αν είναι πιθανό κάποιος συγγραφέας να είναι συν-συγγραφέας με τους συγγραφείς εισόδου. Υπάρχει επίσης η δυνατότητα ταξινόμησης των αποτελεσμάτων βάσει των ‘score’ ή βάσει των κοινών συνέδριων/άρθρων μεταξύ τους.

Το σύστημα αυτό προσφέρει την δυνατότητα εύρεσης κριτών για επιστημονικά άρθρα με αυτοματοποιημένο τρόπο, κάτι το οποίο δεν ήταν δυνατό με κάποιο άλλο ήδη υπάρχον σύστημα. Η αυτοματοποίηση αυτής της διαδικασίας μειώνει τον χρόνο και τον κόπο που θα χρειάζονταν αν αυτή η διαδικασία γινόταν χειροκίνητα από κάποιο άτομο.

1.4 Δομή Εργασίας

Η Διπλωματική Εργασία αποτελείται από τα ακόλουθα κεφάλαια:

Κεφάλαιο 1:
Σε αυτό το κεφάλαιο δίνεται μια γενική εισαγωγή στο θέμα με το οποίο ασχολήθηκα για την Διπλωματική Εργασία. Αναλύεται ο στόχος ο οποίος τέθηκε στην αρχή όταν η εργασία βρισκόταν στα αρχικά της στάδια. Δίνεται επίσης μια γενική περιγραφή της εφαρμογής που αναπτύχθηκε, του τόπου που λειτουργεί, και των λειτουργειών που προσφέρει. Τέλος, δίνεται η δομή της Διπλωματικής Εργασίας και μια σύντομη περιγραφή για τα περιεχόμενα του κάθε κεφαλαίου.

Κεφάλαιο 2:
Στο κεφάλαιο αυτό αρχικά δίνεται μια σύντομη περιγραφή του προβλήματος το οποίο επιλύει το σύστημα που αναπτύχθηκε, και στην συνέχεια ορίζονται κάποιοι όροι, όπως Front-End Development και Back-End Development, και δίνονται περιγραφές των εργαλείων που χρησιμοποιήθηκαν για την ανάπτυξη του, όπως γλώσσες προγραμματισμού και IDEs. Επίσης γίνεται μια γενική αναφορά στα συστήματα συστάσεων και δίνονται κάποια παραδείγματα μεγάλων εταιρειών που τα χρησιμοποιούν. Τέλος, γίνεται αναφορά διάφορων συστημάτων συστάσεων και αναφορά σε αλγόριθμους που συχνά εμφανίζονται σε αυτούς.

Κεφάλαιο 3:
Το κεφάλαιο 3 αφορά ό,τι έχει να κάνει με την ανάπτυξη του συστήματος. Αναφέρεται το Μοντέλο Κύκλου Ζωής που ακολουθήθηκε κατά την διάρκεια της ανάπτυξης τους συστήματος, αλλά και τους περιορισμούς και τους στόχους ευχρηστίας που τέθηκαν για το σύστημα. Αναφέρονται ακόμα και οι απαιτήσεις που τέθηκαν για το σύστημα από τα αρχικά στάδια. Στο τέλος του κεφαλαίου, περιγράφεται η αρχιτεκτονική του συστήματος, και γίνεται περιγραφή των οθονών του.

Κεφάλαιο 4:

Σε αυτό το κεφάλαιο αναφέρονται όλες οι πληροφορίες που έχουν να κάνουν με τον αλγόριθμο που αναπτύχθηκε. Αρχικά δίνεται η βασική ιδέα του αλγόριθμου, και χωρίζεται σε κάποιες βασικές φάσεις. Ακολούθως περιγράφεται και δικαιολογείται ο αλγόριθμος που επιλέχθηκε για το σύστημα και αναφέρεται ο τρόπος χειρισμού των δεδομένων που αντλήθηκαν από το DBLP. Στη συνέχεια περιγράφονται κάποιες δομές που θα αναφερθούν στην περιγραφή. Τέλος, για κάθε φάση του αλγόριθμου δίνεται ψευδοκώδικας και μια εκτενής περιγραφή για την λειτουργία σε εκείνη την φάση.

Κεφάλαιο 5:

Το κεφάλαιο αυτό αφορά την αξιολόγηση του συστήματος. Αναφέρονται διάφορες είσοδοι που δοκιμάστηκαν και παρουσιάζεται γραφική παράσταση που αφορά το χρόνο εκτέλεσης του αλγόριθμου. Τέλος, συζητούνται και αξιολογούνται τα αποτελέσματα όσο αφορά την ποιότητα τους, και παρουσιάζονται κάποια συμπεράσματα.

Κεφάλαιο 6:

Το κεφάλαιο αυτό έχει να κάνει με τα συμπεράσματα της εργασίας μου. Αρχικά γίνεται μια γενική ανασκόπηση όπου αναφέρεται ο στόχος που είχε τεθεί από την αρχή για το σύστημα, διάφορες νέες γνώσεις που έπρεπε να μάθω, αλλά και κάποια εμπόδια που αντιμετώπισα στην πορεία μέχρι την ολοκλήρωση της εργασίας μου. Στην συνέχεια δίνεται το αποτέλεσμα της εργασίας μου. Στο τέλος γίνονται κάποιες εισηγήσεις για μελλοντικές εργασίες.

Κεφάλαιο 2

Συστήματα Συστάσεων - Εργαλεία και Αλγόριθμοι

2.1 Περιγραφή Προβλήματος

Η ιδέα για την διπλωματική μου εργασία προέκυψε μετά από την παρατήρηση πως η διαδικασία επιλογής κριτών για επιστημονικά άρθρα θα μπορούσε να αυτοματοποιηθεί, γλιτώνοντας χρόνο και κόπο στους μέχρι τώρα υπεύθυνους της διαδικασίας αυτής. Η διαδικασία αυτή μέχρι τώρα γίνεται ένα άτομο, που μετά από την μελέτη ενός επιστημονικού άρθρου αλλά και των συγγραφέων αυτού του άρθρου, θέτει τους πιο κατάλληλους κριτές για αυτό το συγκεκριμένο άρθρο βασισμένος σε κάποια κριτήρια.

Η πιο πάνω διαδικασία όμως μπορεί να γίνει χρονοβόρα και αρκετά περίπλοκη σε κάποιες περιπτώσεις, και για τον λόγο αυτό προτάθηκε η ιδέα ανάπτυξης μιας εφαρμογής που θα κάνει τα βήματα επιλογής κριτών αυτοματοποιημένα, κάνοντας όλους τους απαραίτητους ελέγχους και ακολουθώντας τα σωστά κριτήρια. Η επιλογή αυτή γίνεται με βάση τους συγγραφείς ενός άρθρου, αλλά και κάποιες λέξεις-κλειδιά που παρέχονται από τον χρήστη. Με την ανάπτυξη αυτής της εφαρμογής επιλύονται τα προαναφερθέντα προβλήματα, και βεβαιώνεται πως τα αποτελέσματα που θα έχουμε θα είναι τα βέλτιστα.

Στην συνέχεια ακολουθούν περιγραφές των τεχνολογιών που χρησιμοποιήθηκαν κατά τη διάρκεια της εκπόνησης της εργασίας μου.

2.2 Εφαρμογή Ιστού

Μια Εφαρμογή Ιστού είναι μια εφαρμογή η οποία είναι προσβάσιμη μέσω ενός web browser, και φιλοξενείται σε ένα διακομιστή ιστού (web server). Οι Εφαρμογές Ιστού είναι κατασκευασμένες για να παρέχουν στους χρήστες τους διάφορες λειτουργίες στο Διαδίκτυο και δεν είναι αναγκαίο να εγκαθίστανται στον υπολογιστή. Αυτό τις καθιστά πολύ ευέλικτες, αφού οι χρήστες έχουν πρόσβαση σε αυτές από οποιαδήποτε συσκευή που διαθέτει σύνδεση στο Διαδίκτυο και πρόγραμμα περιήγησης ιστού. Στα εργαλεία ανάπτυξης μιας εφαρμογής ιστού συμπεριλαμβάνονται οι HTML, CSS, JavaScript και γλώσσες προγραμματισμού από την πλευρά του server όπως PHP, Ruby και Python.

2.3 Front-End Development

Front-End Development, ή ανάπτυξη από την πλευρά του πελάτη, είναι ο σχεδιασμός και η ανάπτυξη της διεπαφής χρήστη μιας ιστοσελίδας ή εφαρμογής. Δηλαδή είναι η σχεδίαση, ανάπτυξη και υλοποίηση των οπτικών και των διαδραστικών πτυχών μιας ιστοσελίδας με τις οποίες οι χρήστες μπορούν να αλληλεπιδράσουν άμεσα, όπως κουμπιά, μενού, φόρμες και άλλα γραφικά στοιχεία. Για τη διαμόρφωση της δομής αλλά και της αισθητικής μιας ιστοσελίδας ή εφαρμογής, χρησιμοπούνται γλώσσες όπως η HTML και CSS, ενώ για τα στοιχεία που κάνουν την ιστοσελίδα διαδραστική και λειτουργική (π.χ. κουμπιά), χρησιμοποιείται η γλώσσα προγραμματισμού JavaScript.

2.3.1 Γλώσσα Σήμανσης Υπερκειμένου

Η Γλώσσα Σήμανσης Υπερκείμενου (HTML)[3] είναι ένα από τα ποιο σημαντικά στοιχεία του Παγκόσμιου Ιστού και μας επιτρέπει να προσδιορίζουμε τη δομή και την εμφάνισή των ιστοσελίδων που δημιουργούμε, αλλά και να συμπεριλάβουμε διάφορα στοιχεία πολυμέσων σε αυτή. Τα διάφορα στοιχεία μιας ιστοσελίδας όπως επικεφαλίδες, παράγραφοι, σύνδεσμοι, φωτογραφίες, βίντεο και φόρμες, δημιουργούνται χρησιμοποιώντας διαφορετικές ετικέτες στην HTML. Οι δυνατότητες της HTML την καθιστούν ως ένα από τα πιο σημαντικά εργαλεία για τη δημιουργία ιστοσελίδων που θα είναι εύχρηστες για τους χρήστες τους.

2.3.2 JavaScript

Η JavaScript[4] είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου που χρησιμοποιείται για τη δημιουργία δυναμικών διαδικτυακών εφαρμογών και για την προσθήκη

διαδραστικότητας σε αυτές. Μπορεί να χρησιμοποιηθεί τόσο από την πλευρά του server (Node.js), όσο και από την πλευρά του πελάτη. Κάποιες δυναμικές δυνατότητες που παρέχει η JavaScript είναι η κατασκευή αντικειμένων σε χρόνο εκτέλεσης, η δημιουργία δυναμικών σεναρίων κ.α. Με την χρήση της γλώσσας αυτής μπορούν να προστεθούν διάφορες δυναμικές λειτουργίες σε μια ιστοσελίδα, όπως λειτουργικά κουμπιά, δυναμικά μεταβαλλόμενοι πίνακες κ.α. Η JavaScript είναι μια από τις πιο ευρέως χρησιμοποιούμενες γλώσσες προγραμματισμού σήμερα και υποστηρίζεται από όλα τα τρέχοντα προγράμματα περιήγησης ιστού.

2.3.3 CSS

Η CSS (Cascading Style Sheets)[5] είναι ένας απλός μηχανισμός που χρησιμοποιείται για την προσθήκη στυλ σε ιστοσελίδες που είναι σχεδιασμένες με την Γλώσσα Σήμανσης Υπερκειμένου (HTML). Με τη χρήση της CSS στοιχεία μιας ιστοσελίδας όπως γραμματοσειρές, χρώματα, διαστήματα και μεγέθη γραμμάτων μπορούν να καθοριστούν από αυτή. Για αυτό τον λόγο η CSS είναι πολύ σημαντική για την ανάπτυξη ιστοσελίδων και εφαρμογών που είναι φιλικές προς τον χρήστη.

2.3.4 JavaScript Object Notation (JSON)

Το JSON[6] είναι μια ελαφριά μορφή ανταλλαγής δεδομένων που χρησιμοποιείται για ανταλλαγή δεδομένων μεταξύ ενός διακομιστή (server) και ενός client. Η μορφή του είναι text-based, δηλαδή είναι βασισμένη σε κείμενο και έτσι είναι εύκολο για τους ανθρώπους να τη διαβάζουν και να τη γράφουν. Είναι επίσης εύκολο για τις μηχανές να την αναλύουν και να τη δημιουργούν. Η βάση του JSON είναι ένα σύνολο ζευγών ονομάτων-τιμών, όπου κάθε όνομα υποδηλώνει μια συμβολοσειρά και κάθε τιμή μπορεί να είναι συμβολοσειρά, αριθμός, αντικείμενο, πίνακας, boolean, ή null. Στις διάφορες γώσσες προγραμματισμού ένα αντικείμενο JSON μπορεί να αναπαριστάται ως πίνακας, ως λίστα, ως μια ακολουθία ή ως ένα αντικείμενο. Επειδή ένα αντικείμενο JSON μπορεί εύκολα να υποβληθεί σε επεξεργασία και να μετατραπεί σε αντικείμενο στην JavaScript, δίνοντας τη δυνατότητα για δυναμική τροποποίηση του περιεχομένου μιας ιστοσελίδας χωρίς να χρειάζεται να φορτωθεί ξανά ολόκληρη η σελίδα, χρησιμοποιείται συχνά για την επικοινωνία και την ανταλλαγή δεδομένων μεταξύ διακομιστή και εφαρμογής ιστού.

2.4 Back-End Development

Το Back-End Development ή Server Side Development, είναι η διαδικασία δημιουργίας και συντήρησης του διακομιστή, της βάσης δεδομένων αλλά και της λογικής που τροφοδοτούν το Front-End μιας ιστοσελίδας ή εφαρμογής. Η λογική και η λειτουργικότητα που χρειάζεται δημιουργούνται με την χρήση διαφόρων γλωσσών προγραμματισμού όπως PHP, Python, Ruby, και Java. Επίσης γίνεται αλληλεπίδραση με βάσεις δεδομένων όπως MySQL, Oracle, και MongoDB για τον έλεγχο και την συντήρηση των δεδομένων των οποίων φυλάγονται σε αυτές.

2.4.1 PHP

Η PHP[7] είναι μια ανοιχτού κώδικα γλώσσα δέσμης ενεργειών που είναι κατάλληλη για να χρησιμοποιηθεί στο Web Development. Μπορεί να συμπεριληφθεί στην HTML, και ο κώδικας γραμμένος σε PHP τρέχει στην πλευρά του διακομιστή (server-side). Επιτρέπει τη δημιουργία στατικών, δυναμικών, και περίπλοκων δυναμικών ιστοσελίδων και εφαρμογών ιστού, που αλληλεπιδρούν με βάσεις δεδομένων κλπ. Η PHP είναι η επικρατέστερη επιλογή μεταξύ των προγραμματιστών ιστού (web developers), λόγω της απλότητας και της προσαρμοστικότητας της.

2.4.2 Python

Η Python[8] είναι μια υψηλού επιπέδου αντικειμενοστραφής γλώσσα προγραμματισμού που χρησιμοποιείται συχνά για εργασίες που περιλαμβάνουν web development, επιστημονική πληροφορική (scientific computing), ανάλυση δεδομένων (data analysis), τεχνητή νοημοσύνη (artificial intelligence), μηχανική μάθηση (machine learning), κ.α. Ο τρόπος σύνταξης που χρησιμοποιεί αλλά και η ευχρηστία της κάνουν τα προγράμματα γραμμένα σε αυτή τη γλώσσα πιο ευανάγνωστα και διευκολύνουν τη λειτουργία τους. Έχει διαθέσιμη μια μεγάλη βιβλιοθήκη με modules, τα οποία δίνουν τη δυνατότητα για διάφορες λειτουργίες όπως σύνδεση σε διακομιστές, ανάπτυξη ιστοσελίδων και επεξεργασία δεδομένων, και έχει μια ζωντανή κοινότητα προγραμματιστών που συμβάλλουν συνέχεια για την ανάπτυξη αυτής της βιβλιοθήκης, και επομένως στην αύξηση των δυνατοτήτων της γλώσσας. Λόγω του ότι η Python είναι απλή, καθίσταται μια κατάλληλη γλώσσα για αρχάριους προγραμματιστές, που θα τους βοηθήσει να καταλάβουν τις βασικές αρχές του προγραμματισμού.

2.5 Integrated Development Environment (IDE)

Ένα IDE (Integrated Development Environment), είναι ένα πρόγραμμα λογισμικού που προσφέρει εκτεταμένες δυνατότητες για ανάπτυξη λογισμικού (software development). Ένα τέτοιο λογισμικό συνήθως περιλαμβάνει επεξεργαστή κώδικα (code editor), απασφαλματωτή (debugger), μεταγλωττιστή (compiler), διερμηνέα (interpreter), και άλλα εργαλεία που προσφέρουν την δυνατότητα αυτοματοποίησης των εργασιών συγγραφής κώδικα, όπως είναι η συμπλήρωση κώδικα (code completion), η μορφοποίηση κώδικα (code formatting), και ο έλεγχος έκδοσης (version control).

2.5.6 JupyterLab

Το JupyteLab[9] είναι μια web-based διεπαφή χρήστη για το Project Jupyter. Το JupyterLab παρέχει την δυνατότητα στους χρήστες του να δουλεύουν με έγγραφα, notebooks, επεξεργαστές κειμένου κ.α., σε ένα ευέλικτο περιβάλλον. Τους επιτρέπει επίσης να γράφουν κώδικα σε γλώσσες όπως Python και R, να επιθεωρούν και να αλλάζουν δεδομένα και να εμφανίζουν τα δεδομένα σε ένα ενιαίο περιβάλλον. Επιπρόσθετα, παρέχει την δυνατότητα της προσθήκης διάφορων επεκτάσεων (extensions), οι οποίες μπορούν να προσθέσουν νέες δυνατότητες και εργαλεία.

2.5.7 Visual Studio Code

Το Visual Studio Code[10] είναι ένα ελαφρύ και ισχυρό πρόγραμμα επεξεργασίας κώδικα που προσφέρει την δυνατότητα επεξεργασίας κώδικα, αλλά και συμπλήρωσης κώδικα, επισήμανσης σύνταξης, εντοπισμού σφαλμάτων, και ενσωμάτωσης Git. Μερικές από τις γλώσσες προγραμματισμού που υποστηρίζονται από το Visual Code Studio με την χρήση επεκτάσεων για αυτές είναι η Java, η Python, η HTML, η JavaScript κ.α. Οι διάφορες επεκτάσεις που είναι διαθέσιμες στο Visual Studio Code δίνουν επίσης τη δυνατότητα για προσθήκη νέων δυνατοτήτων στις ήδη υπάρχουσες.

2.6 Συστήματα Συστάσεων

Τα συστήματα συστάσεων (recommender systems)[11] χρησιμοποιούν τεχνητή νοημοσύνη (Artificial Intelligence) και σχεδιάζονται με σκοπό να προβλέπουν και να προτείνουν στοιχεία τα οποία είναι πιθανότερο να ενδιαφέρουν τον χρήστη, με βάση προηγούμενες

συμπεριφορές ή προτιμήσεις του, ή με κάποια άλλα σχετικά δεδομένα. Η ανάπτυξη και εφαρμογή ενός τέτοιου συστήματος για οποιοδήποτε σκοπό μπορεί να αποτελέσει πρόκληση λόγω κάποιων προκλήσεων που μπορούν να εμφανιστούν. Μία από αυτές είναι η αντιμετώπιση της αραιότητας δεδομένων, δηλαδή όταν δεν υπάρχουν αρκετά διαθέσιμα δεδομένα έτσι ώστε να μπορεί το σύστημα να τα χρησιμοποιήσει και να δώσει ορθές και ακριβείς συστάσεις. Αυτό μπορεί να γίνει λόγω πρόσθεσης νέων αντικειμένων σε κατάλογο, για παράδειγμα, για το οποίο δεν υπάρχουν προηγούμενα στοιχεία. Μια άλλη πρόκληση που μπορεί να παρουσιαστεί, είναι η επιλογή του κατάλληλου αλγόριθμου ή συνδυασμού αλγορίθμων για το σύστημα συστάσεων. Αυτό εξαρτάται από τον σκοπό του κάθε συστήματος συστάσεων και από τα δεδομένα που θα πρέπει να επεξεργαστεί. Τέλος, ακόμα ένα θέμα που μπορεί να εμφανιστεί έχει να κάνει με την ιδιωτικότητα των χρηστών. Η παροχή προσωπικών δεδομένων σε σύστημα συστάσεων έχει ως αποτέλεσμα καλύτερες συστάσεις προς τον χρήστη. Αν όμως το σύστημα έχει θέματα με την ιδιωτικότητα των δεδομένων, οι χρήστες δεν είναι πρόθυμοι να δίνουν τα δεδομένα τους. Για αυτό τέτοια συστήματα θα πρέπει να λαμβάνουν μέτρα έτσι ώστε, αν χρειάζεται οι χρήστες να καταχωρούν προσωπικά τους δεδομένα, να είναι σίγουρο πως θα είναι ασφαλή. Τέτοια συστήματα έχουν υλοποιηθεί από πάρα πολλές μεγάλες εταιρείες, και σε κάθε περίπτωση χρησιμοποιούνται στα πλαίσια αυτών. Κάποια πολύ γνωστά παραδείγματα είναι τα ακόλουθα:

Amazon

Η Amazon[12] χρησιμοποιεί σύστημα συστάσεων έτσι ώστε να προτείνει προϊόντα στους πελάτες τους με βάση άλλα προϊόντα που είδαν ή αγόρασαν, αλλά και με κοινά χαρακτηριστικά που μπορεί να έχουν με άλλους πελάτες.

Netflix

Το Netflix[13] χρησιμοποιεί σύστημα συστάσεων με σκοπό να προτείνει στους χρήστες του ταινίες και σειρές βάσει προηγούμενων ταινιών και σειρών που είδαν αλλά και με τις βαθμολογήσεις που έδωσαν σε αυτές.

Spotify

Το Spotify[14] χρησιμοποιεί σύστημα συστάσεων έτσι ώστε να προτείνει τραγούδια και καλλιτέχνες στους χρήστες του με βάση τα τραγούδια και τους καλλιτέχνες που ακούν συνήθως.

Τύποι συστημάτων συστάσεων

Content-based filtering [15]:

Αυτός ο τύπος συστημάτων συστάσεων προτείνει στον χρήστη στοιχεία με βάση την συσχέτιση των περιεχομένων αυτών των στοιχείων, και των προτιμήσεων του χρήστη. Ένας αλγόριθμος που περιλαμβάνονται συνήθως σε ένα τέτοιο σύστημα συστάσεων είναι ο TF-IDF [16].

Collaborative filtering [17]:

Αυτός ο τύπος συστημάτων συστάσεων προτείνει στο χρήστη στοιχεία συνδέοντας τις προτιμήσεις που είχε ο χρήστης στο παρελθόν με τις προτιμήσεις άλλων χρηστών. Οι αποφάσεις για το φιλτράρισμα των στοιχείων γίνονται με βάση ανθρώπους και τις προτιμήσεις τους και όχι με βάση αναλύσεων από μηχανές, όπως γίνεται στο content-based filtering. Αυτή η διαφορά δίνει και κάποια πλεονεκτήματα στο collaborative filtering όπως είναι το γεγονός ότι έχει την δυνατότητα να φιλτράρει οπουδήποτε είδος περιεχομένου. Σε πολλές περιπτώσεις content-based filtering και collaborative filtering μπορούν να χρησιμοποιηθούν σε συνδυασμό έτσι ώστε να δημιουργηθεί ένα ισχυρό υβριδικό σύστημα. Ένας αλγόριθμος που περιλαμβάνονται συχνά σε ένα τέτοιο σύστημα συστάσεων είναι ο k-Nearest-Neighbors[2].

Hybrid Recommender Systems:

Αυτά τα συστήματα συστάσεων εφαρμόζουν ένα συνδυασμό από δύο ή περισσότερα άλλα είδη συστημάτων συστάσεων με σκοπό να παρέχουν πιο ακριβή και στοχευμένα αποτελέσματα. Αλγόριθμοι που περιλαμβάνονται σε τέτοια συστήματα μπορεί να είναι οποιοσδήποτε συνδυασμός αλγορίθμων από τους υπόλοιπους τύπους συστημάτων συστάσεων.

Context-Aware Recommender Systems [18]:

Αυτά τα συστήματα λαμβάνουν υπόψη παράγοντες όπως την ώρα, την τοποθεσία, και τις καιρικές συνθήκες, με σκοπό να παρέχει πιο εξατομικευμένα και σχετικά αποτελέσματα. Αλγόριθμοι που περιλαμβάνονται σε τέτοια συστήματα μπορεί να είναι οποιοσδήποτε συνδυασμός αλγορίθμων από τους υπόλοιπους τύπους συστημάτων συστάσεων, αλλά και αλγόριθμοι που να ενσωματώνουν παράγοντες όπως αυτοί που αναφέρθηκαν πιο πάνω.

Κεφάλαιο 3

Μοντέλο Κύκλου Ζωής, Απαιτήσεις και Ανάπτυξη Συστήματος

3.1 Μοντέλο Κύκλου Ζωής

Για την ανάπτυξη ενός συστήματος όπως το δικό μας, το καταλληλότερο Μοντέλο Κύκλου Ζωής είναι οι μεθοδολογία Agile[19]. Η μεθοδολογία αυτή είναι ουσιαστικά μια επαναληπτική και σταδιακή προσέγγιση, όπου δίνεται έμφαση στη συνεργασία με τους χρήστες στην γρήγορη παράδοση ενός λειτουργικού λογισμικού. Κάθε επανάληψη δίνει το επόμενο λειτουργικό κομμάτι του λογισμικού και είναι διαθέσιμο για χρήση από τους ενδιαφερόμενους μέχρι να ολοκληρωθεί η ανάπτυξη του συστήματος. Επίσης, σε κάθε επανάληψη οι χρήστες παρέχουν ανατροφοδότηση , πράγμα που βοηθά τον γρήγορο εντοπισμό και διόρθωση σφαλμάτων. Για αυτούς τους λόγους κρίθηκε ως η πιο κατάλληλη μεθοδολογία για την ανάπτυξη ενός συστήματος όπως είναι το Σύστημα Συστάσεων το οποίο αναπτύσσουμε. Με τη συχνή και συνεχή παρουσίαση λειτουργικού λογισμικού, δίνεται επίσης η δυνατότητα στους χρήστες να αλληλεπιδρούν με το σύστημα, και έτσι να έχουν μια καλύτερη εικόνα για το σύστημα και για τις αλλαγές ή και επιπρόσθετες λειτουργίες που θέλουν να δουν στο σύστημα στο μέλλον.

3.2 Περιορισμοί Συστήματος

Για να είναι προσβάσιμο το σύστημα μας από όλους, απαιτείται ένας web server ο οποίος θα φιλοξενεί την ιστοσελίδα του συστήματος. Ο εν λόγω server θα πρέπει να έχει εγκατεστημένη την Python 3 και να μπορούν να εγκατασταθούν διάφορα modules της σε αυτόν, έτσι ώστε να μπορεί να τρέχει ο αλγόριθμος (γραμμένος σε Python) στο backround.

Οι χρήστες του συστήματος μας, για να έχουν πρόσβαση σε αυτό θα πρέπει να έχουν στη διάθεση τους έναν ηλεκτρονικό υπολογιστή, έξυπνο τηλέφωνο, ή tablet, τα οποία να είναι συνδεδεμένα με το Διαδίκτυο και να έχουν διαθέσιμο κάποιο φυλλομετρητή (web browser).

3.3 Στόχοι Ευχρηστίας Συστήματος

3.3.1 Ορισμός Στόχων Ευχρηστίας

Πρωταρχικός στόχος για ένα σύστημα αποτελεί η ευχρηστία του. Είναι η κύρια παράμετρος με την οποία αξιολογείται η ποιότητα ενός συστήματος και είναι αναγκαία για την ορθή ανάπτυξη του με επίκεντρο τις ανάγκες του χρήστη. Η ανθρωποκεντρική σχεδίαση σε συνδυασμό με την ευχρηστία οδηγούν σε ένα σύστημα το οποίο είναι αποτελεσματικό, αποδοτικό, φιλικό προς τον χρήστη, και δίνει ευχαρίστηση στον χρήστη όταν το χρησιμοποιεί.
Η ευχρηστία του συστήματος επιτυγχάνεται ακολουθώντας τις πέντε παραμέτρους του J.Nielsen:

1. Ευκολία και ταχύτητα εκμάθησης χρήσης του συστήματος
2. Υψηλή απόδοση εκτέλεσης των λειτουργιών
3. Ικανότητα συγκράτησης της ικανότητας χρήσης του συστήματος με την πάροδο του χρόνου
4. Μικρός αριθμός εσφαλμένων χειρισμών και εύκολος τρόπος ανάνηψης από αυτά
5. Υποκειμενική ικανοποίηση των χρηστών από την επαφή τους με το σύστημα.

3.3.2 Τεκμηρίωση Στόχων Ευχρηστίας

1. **<u>Ευκολία και ταχύτητα εκμάθησης χρήσης του συστήματος</u>**

   Είναι μεγάλης σημασίας οι χρήστες του συστήματος να μπορούν να το χρησιμοποιούν με ευκολία και χωρίς να καταβάλλουν ιδιαίτερα μεγάλη προσπάθεια, όπως επίσης και να μην τους είναι απαραίτητη κάποιου είδους καθοδήγηση. Πρέπει επίσης να στοχεύσουμε στην εύκολη προσαρμογή του χρήστη με το σύστημα. Για αυτό τον λόγο χρησιμοποιήθηκαν εικονίδια και κουμπιά που είναι επεξηγηματικά ως προς την λειτουργία τους, και δεν προκαλούν κάποια σύγχυση στον χρήστη. Επιπρόσθετα, ο σχεδιασμός της σελίδας είναι αρκετά απλός έτσι ώστε να δίνει την

μέγιστη ικανοποίηση στον χρήστη, και ο τρόπος λειτουργίας του συστήματος είναι πολύ απλός και εύκολος.

2. **Υψηλή απόδοση εκτέλεσης των λειτουργιών**

Οι χρήστες θα μπορούν να εκτελούν τις λειτουργίες της εφαρμογής αποδοτικά, δηλαδή το σύστημα θα εκπληρώνει όλους τους στόχους του χρήστη. Στόχος είναι ο κάθε χρήστης να είναι ικανοποιημένος αφού χρησιμοποιήσει το σύστημα. Για αυτό τον λόγο δόθηκε έμφαση στην βελτίωση του χρόνου που χρειάζεται για να τρέξει ο αλγόριθμος, αλλά και στην απλότητα που πρέπει να υπάρχει στην διαδικασία εκτέλεσης του.

3. **Ικανότητα συγκράτησης της ικανότητας χρήσης του συστήματος με την πάροδο του χρόνου**

Βασικός στόχος είναι η εμπειρία των χρηστών του συστήματος να είναι καλύτερη κάθε φορά που αλληλεπιδρούν με αυτό. Μετά την πρώτη τους επαφή με το σύστημα, δεν θα πρέπει να χρειάζονται κάποια βοήθεια για να επαναλάβουν τις λειτουργίες. Για αυτό τον λόγο χρησιμοποιούνται επεξηγήσεις σε όποιο σημείο ο χρήστης πρέπει να εισάγει κάτι ή για την λειτουργικότητα των διάφορων κουμπιών. Η γενική απλότητα του συστήματος καθιστά εύκολη την προσαρμογή του κάθε χρήστη.

4. **Μικρός αριθμός εσφαλμένων χειρισμών και εύκολος τρόπος ανάνηψης από αυτά**

Στα σημεία όπου απαιτείται από τον χρήστη να εισάγει πληροφορίες, έχουμε φροντίσει έτσι ώστε να εμφανίζονται μηνύματα λάθους έτσι ώστε ο χρήστης να ενημερώνεται και ταυτόχρονα να καθοδηγείται στο να εκτελέσει σωστά την ενέργεια που επιθυμεί.

5. **Υποκειμενική ικανοποίηση των χρηστών από την επαφή τους με το σύστημα.**

Ο χρήστης πρέπει, με την αλληλεπίδραση του με το σύστημα, να λαμβάνει μια υποκειμενική ευχαρίστηση. Αυτό επιτυγχάνεται με την αποδοτική και αποτελεσματική εκπλήρωση των ενεργειών του καθώς και με τη συνεχή βελτίωση και έλεγχο του συστήματος που θα γίνεται πάντα σύμφωνα με τις ανάγκες του χρήστη.

3.4 Απαιτήσεις Συστήματος

3.4.1 Απαιτήσεις για τους Χρήστες

i. Να μπορεί να εισάγει όσα ονόματα συγγραφέων χρειάζεται.
ii. Να μπορεί να εισάγει όσες λέξεις κλειδιά χρειάζεται.
iii. Να μπορεί να καθορίσει τον αριθμό των αποτελεσμάτων που θα εμφανιστούν στο τέλος.
iv. Να μπορεί να δει και να καταλάβει τα αποτελέσματα του αλγορίθμου

3.4.2 Λειτουργικές Απαιτήσεις

i. Εκτέλεση του αλγόριθμου χρησιμοποιώντας ως κριτήριο τα κοινά άρθρα συνεδρίων στα οποία συνέβαλαν οι συγγραφείς.

   Ο χρήστης μπορεί να εκτελέσει τον αλγόριθμο με κριτήριο τα κοινά άρθρα συνεδρίων στα οποία συνέβαλαν οι συγγραφείς, με τη χρήση του αντίστοιχου κουμπιού.

ii. Εκτέλεση του αλγόριθμου χρησιμοποιώντας ως κριτήριο τα κοινά άρθρα περιοδικών μεταξύ των συγγραφέων.

   Ο χρήστης μπορεί να εκτελέσει τον αλγόριθμο με κριτήριο τα κοινά άρθρα περιοδικών μεταξύ των συγγραφέων, με τη χρήση του αντίστοιχου κουμπιού.

iii. Εκτέλεση του αλγόριθμου χρησιμοποιώντας ως κριτήριο ένα συνδυασμό μεταξύ των κοινών άρθρων συνεδρίων στα οποία συνέβαλαν οι συγγραφείς, και των κοινών άρθρων περιοδικών μεταξύ των συγγραφέων.

   Ο χρήστης μπορεί να εκτελέσει τον αλγόριθμο με κριτήριο τον συνδυασμό των κοινών άρθρων συνεδρίων στα οποία συνέβαλαν οι συγγραφείς και των κοινών άρθρων περιοδικών μεταξύ τους, με τη χρήση του αντίστοιχου κουμπιού.

iv. Εκκαθάριση εισόδων από πληροφορίες που έχουν γραφτεί και εκκαθάριση αποτελεσμάτων.

Ο χρήστης μπορεί να καθαρίσει όλες τις εισόδους που έχει δώσει, και επίσης να εξαφανίσει τον πίνακα στον οποίο παρουσιάζονται τα αποτελέσματα του αλγόριθμου με την χρήση του αντίστοιχου κουμπιού.

v. Ταξινόμηση των αποτελεσμάτων με βάση τον αριθμό των λέξεων κλειδιών που βρέθηκαν να συνδέονται με κάθε συγγραφέα.

Ο χρήστης μπορεί να ταξινομήσει τα αποτελέσματα που εμφανίζονται στον πίνακα αποτελεσμάτων του αλγόριθμου, με το πάτημα της κεφαλίδας της ανάλογης στήλης (στήλη με σκορ λέξεων κλειδιών).

vi. Ταξινόμηση των αποτελεσμάτων με βάση τον αριθμό των κοινών άρθρων συνεδρίων/άρθρων περιοδικών που βρέθηκαν για κάθε συγγραφέα.

Ο χρήστης μπορεί να ταξινομήσει τα αποτελέσματα που εμφανίζονται στον πίνακα αποτελεσμάτων του αλγόριθμου, με το πάτημα της κεφαλίδας της ανάλογης στήλης (στήλη με κοινά Συνέδρια/Περιοδικά) όπου υπάρχει και το αντίστοιχο εικονίδιο που τον βοηθά να καταλάβει τη συγκεκριμένη λειτουργία.

3.4.3 Μη Λειτουργικές Απαιτήσεις

i. Ευχρηστία

Το σύστημα πρέπει να ακολουθεί όλους τους κανόνες ευχρηστίας, όπως αναφέρθηκαν πιο πάνω, έτσι ώστε να παρέχετε στον χρήστη η μέγιστη ευχαρίστηση και ικανοποίηση όταν το χρησιμοποιεί.

ii. Αποδοτικότητα

Το σύστημα θα πρέπει να εκτελεί τις λειτουργίες του σε σχετικά γρήγορο χρόνο έτσι ώστε να αποφευχθεί οποιαδήποτε δυσφορία από τον χρήστη.

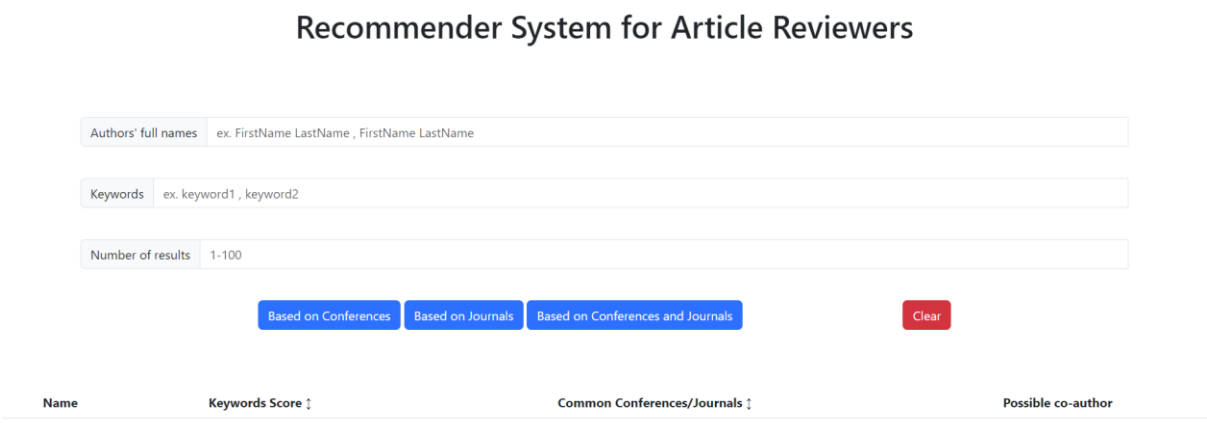
iii. Αξιοπιστία

Το σύστημα θα πρέπει να χρησιμοποιεί σωστά τις εισόδους που του παρέχει ο χρήστης και να παρέχει ορθά αποτελέσματα, τα οποία πραγματικά να βασίζονται σε αυτές τις εισόδους.

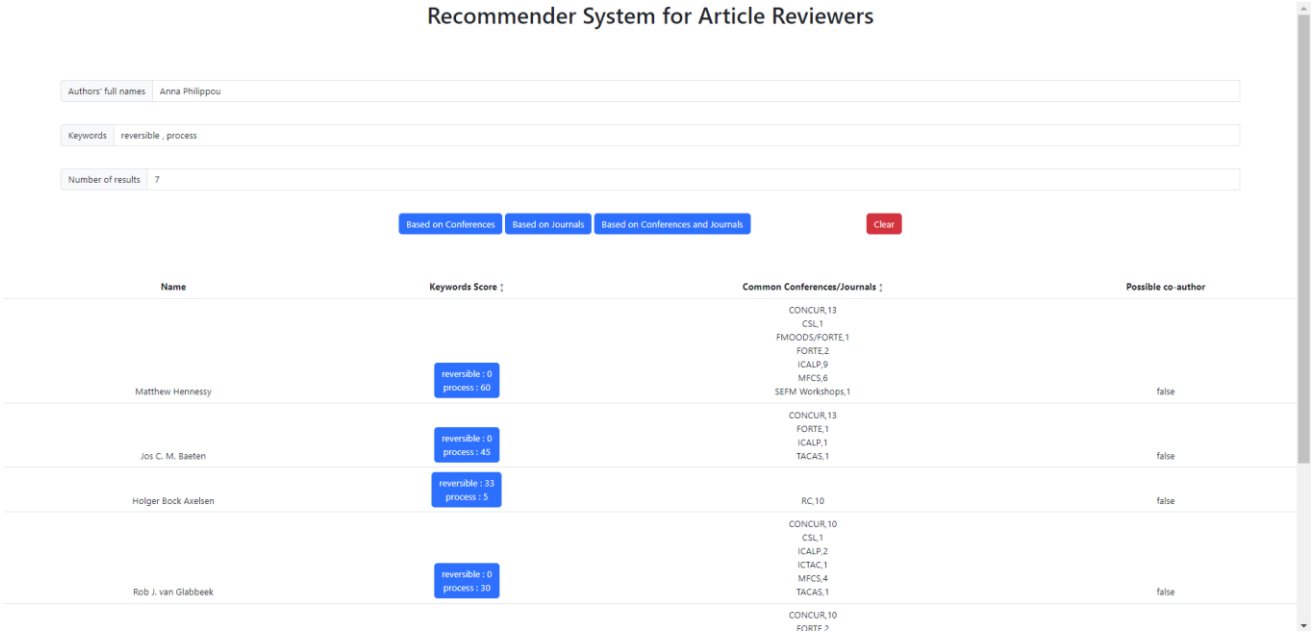
3.5 Αρχιτεκτονική Συστήματος

Το σύστημα μας αποτελείται από δύο βασικά στοιχεία, όπου το πρώτο στοιχείο είναι το web application. Το web application χωρίζεται σε δύο κομμάτια, το Front-End και το Back-End. Το Front-End είναι υπεύθυνο για την δομή (HTML), την αισθητική (CSS), και την λειτουργικότητα (JavaScript) της σελίδας, ενώ το Back-End (PHP) είναι υπεύθυνο για την εκτέλεση του αλγόριθμου με τις εισόδους που πάρθηκαν από το Front-End. Το δεύτερο βασικό στοιχείο του συστήματος είναι ο αλγόριθμος που κάνει την βασική δουλειά του συστήματος, δηλαδή την εύρεση των πιο κατάλληλων κριτών με βάση κάποια κριτήρια. Ο αλγόριθμος αυτός είναι γραμμένος στην γλώσσα προγραμματισμού Python, και καλείται από το Back-End, δηλαδή από την PHP, με τα ορίσματα που πρέπει, και τρέχει στο παρασκήνιο.

3.6 Περιγραφή Οθονών



Εικόνα 3.1 Αρχική Οθόνη



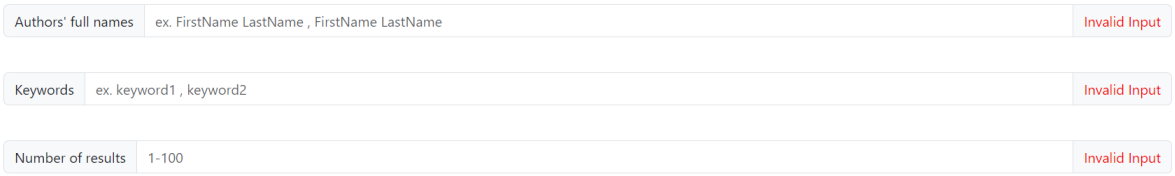
Εικόνα 3.2 Οθόνη Αποτελεσμάτων

3.6.1 Περιγραφή Αρχικής Οθόνης

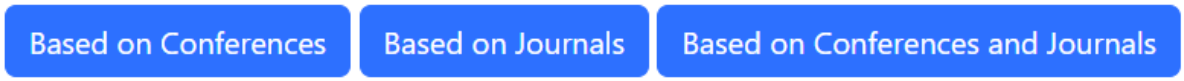
Η οθόνη στην Εικόνα 3.1 εμφανίζεται στον χρήστη όταν επισκεφτεί την ιστοσελίδα του συστήματος. Σε αυτή τη σελίδα, ο χρήστης μπορεί να εισάγει τα στοιχεία με τα οποία θέλει να εκτελεστεί ο αλγόριθμος εύρεσης κατάλληλων κριτών για ένα άρθρο. Αυτές οι είσοδοι είναι τα ονόματα των συγγραφέων του άρθρου για το οποίο θέλει να βρεθούν κριτές, λέξεις κλειδιά που αφορούν το θέμα του άρθρου και θα ήταν βοηθητικές για την εύρεση κριτών για το συγκεκριμένο θέμα, και ο αριθμός των αποτελεσμάτων που θέλει να εμφανιστούν στον πίνακα αποτελεσμάτων όταν ο αλγόριθμος επιστρέψει τα αποτελέσματα. Οι είσοδοι αυτοί θα πρέπει να δοθούν από το χρήστη σε συγκεκριμένη μορφή, όπως αναγράφεται και στις οδηγίες που βρίσκονται στα πεδία εισόδου (Εικόνα 3.3). Σε περίπτωση που ο χρήστης δεν εισάγει κάποιο κείμενο στα πεδία εισόδου, ή δώσει κείμενο εισόδου σε λανθασμένη μορφή και πατήσει κουμπί για να ξεκινήσει τον αλγόριθμο, εμφανίζεται μήνυμα λάθους δεξιά από το πεδίο εισόδου στο οποίο έγινε το λάθος (Εικόνα 3.4). Όταν ο χρήστης εισάγει τις απαραίτητες εισόδους στα πεδία εισόδου, μπορεί να χρησιμοποιήσει ένα από τα 3 μπλε κουμπιά (Εικόνα 3.5) για να ξεκινήσει τον αλγόριθμο. Κάθε κουμπί αντιστοιχεί σε διαφορετική έκδοση του αλγόριθμου, όπου η διαφορά μεταξύ τους είναι η βιβλιογραφική πηγή που θα χρησιμοποιήσει ως κριτήριο ο αλγόριθμος. Με το πρώτο κουμπί, ο αλγόριθμος θα λάβει ως κριτήριο τα συνέδρια στα οποία συνέβαλαν οι συγγραφείς, ενώ με το δεύτερο κουμπί ο αλγόριθμος θα λάβει ως κριτήριο τα άρθρα στα οποία έγραψαν οι συγγραφείς. Όταν ο χρήστης πατήσει ένα κουμπί για να εκκινήσει τον αλγόριθμο, αυτό το κουμπί αλλάζει χρώμα (Εικόνα 3.6), μένοντας έτσι καθ' όλη τη διάρκεια που ο αλγόριθμος τρέχει στο

παρασκήνιο, και επανέρχεται στο αρχικό του χρώμα όταν ο αλγόριθμος τελειώσει. Αυτό γίνεται για να υπάρχει ένδειξη προς τον χρήστη για το πότε τρέχει ο αλγόριθμος. Τέλος, με το τρίτο κουμπί ο αλγόριθμος θα λάβει ως κριτήριο ένα συνδυασμό των συνεδρίων που συνέβαλαν οι συγγραφείς, και των άρθρων τα οποία έγραψαν. Αν ο χρήστης επιθυμεί, έχει επίσης την δυνατότητα να καθαρίσει όλα τα πεδία εισόδου από αυτά που έχουν γραμμένα μέσα, αλλά και τον πίνακα των αποτελεσμάτων αν έχει εμφανιστεί, με το πάτημα του κόκκινου κουμπιού “Clear” (Εικόνα 3.7). Στο κάτω μέρος της οθόνης υπάρχουν οι κεφαλίδες του πίνακα αποτελεσμάτων, μέσα στον οποίο θα εμφανιστούν τα αποτελέσματα με το τέλος του αλγόριθμου.

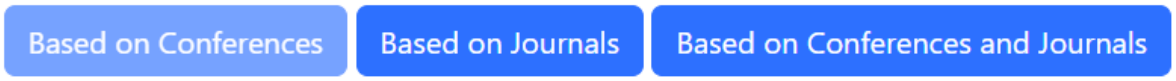
Εικόνα 3.3 Οδηγίες στα πεδία εισόδου



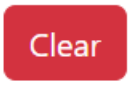
Εικόνα 3.4 Πεδία εισόδου με μηνύματα λάθους στα δεξιά



Εικόνα 3.5 Κουμπιά για εκκίνηση αλγόριθμου



Εικόνα 3.6 Κουμπιά όταν ο αλγόριθμος τρέχει στο παρασκήνιο



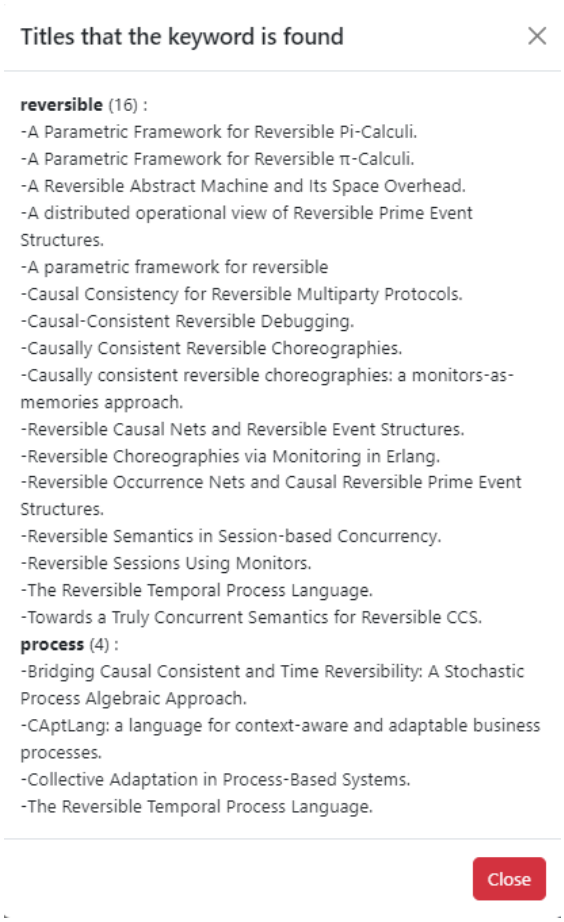
Εικόνα 3.7 Κουμπί για εκκαθάριση πεδίων εισόδου και εξαφάνιση του πίνακα αποτελεσμάτων

3.6.2 Περιγραφή Οθόνης Αποτελεσμάτων

Η οθόνη στην Εικόνα 3.2 εμφανίζεται στον χρήστη αφού τρέξει ο αλγόριθμος και επιστρέψει τα αποτελέσματα του. Ουσιαστικά, τα στοιχεία αυτής της οθόνης δεν διαφέρουν καθόλου από τα στοιχεία που υπάρχουν και στην αρχική οθόνη (Εικόνα 3.1), εκτός από τον πίνακα των αποτελεσμάτων ο οποίος γεμίζει με τα αποτελέσματα του αλγόριθμου (Εικόνα 3.8). Ο αριθμός των αποτελεσμάτων που θα εμφανιστούν εξαρτάται από την είσοδο που είχε δώσει ο χρήστης στην αρχή. Ο πίνακας δίνει τα εξής στοιχεία για κάθε έναν από τους συγγραφείς που προτείνεται: το όνομα του και το σκορ της κάθε λέξης κλειδί για αυτόν, που αντιστοιχεί στον αριθμό των μη διπλότυπων τίτλων των καταχωρήσεων του στη βάση δεδομένων του DBLP που περιέχουν την κάθε λέξη κλειδί. Στην τρίτη στήλη του πίνακα εμφανίζονται όλα τα κοινά συνέδρια ή/και άρθρα μεταξύ του συγκεκριμένου συγγραφέα και των συγγραφέων που έδωσε ως είσοδο ο χρήστης, και στην τελευταία στήλη δίνεται η τιμή true αν υπάρχει περίπτωση ο συγκεκριμένος συγγραφέας να είναι συν-συγγραφέας με κάποιον από τους συγγραφείς που δόθηκαν ως είσοδος, ή δίνεται η τιμή false στην αντίθετη περίπτωση. Ο χρήστης έχει την δυνατότητα να ταξινομήσει τα αποτελέσματα στον πίνακα με βάση είτε το σκορ των λέξεων κλειδιών, είτε τον αριθμό των κοινών συνεδρίων ή/και άρθρων. Αυτό γίνεται με το πάτημα του ποντικιού στην κεφαλίδα της αντίστοιχης στήλης που αφορά την τιμή με την οποία θέλουμε να γίνει η ταξινόμηση. Δηλαδή, για να ταξινομηθεί ο πίνακας με βάση το σκορ των λέξεων κλειδιών, πρέπει να πατηθεί η κεφαλίδα “Keywords Score” στον πίνακα. Αντίστοιχα, για να ταξινομηθεί ο πίνακας με βάση τον αριθμό των κοινών συνέδριων ή/και άρθρων, θα πρέπει να πατηθεί η κεφαλίδα “Common Conferences/Journals” του πίνακα αποτελεσμάτων. Σε περίπτωση που ο χρήστης θέλει να δει με περισσότερη λεπτομέρεια τα σημεία στα οποία βρέθηκαν και βασίστηκε το σκορ των λέξεων κλειδιών, έχει την δυνατότητα να το κάνει με το πάτημα του κουμπιού που βρίσκεται στην δεύτερη στήλη του πίνακα σε κάθε αποτέλεσμα. Με το πάτημα ενός από αυτά τα κουμπιά, εμφανίζεται ένα παράθυρο που δείχνει αναλυτικά όλους τους τίτλους που αφορούν τον συγκεκριμένο συγγραφέα και περιέχουν κάποια από τις λέξεις κλειδιά (Εικόνα 3.9). Κάθε λέξη κλειδί εμφανίζεται με bold γράμματα, δίπλα από αυτή εμφανίζεται το σκορ της, και πιο κάτω εμφανίζονται αναλυτικά όλοι οι τίτλοι που περιέχουν την αντίστοιχη λέξη κλειδί.

Name	Keywords Score ↕	Common Conferences/Journals ↕	Possible co-author
Matthew Hennessy	reversible : 0 process : 60	CONCUR,13 CSL,1 FMOODS/FORTE,1 FORTE,2 ICALP,9 MFCS,6 SEFM Workshops,1	false
Jos C. M. Baeten	reversible : 0 process : 45	CONCUR,13 FORTE,1 ICALP,1 TACAS,1	false
Holger Bock Axelsen	reversible : 33 process : 5	RC,10	false
Rob J. van Glabbeek	reversible : 0 process : 30	CONCUR,10 CSL,1 ICALP,2 ICTAC,1 MFCS,4 TACAS,1	false
Jan Friso Groote	reversible : 0 process : 26	CONCUR,10 FORTE,2 ICALP,3 ICTAC,2 IPDPS,3 MFCS,1 TACAS,4	false
Michael Kirkedal Thomsen	reversible : 23 process : 2	RC,10	false
Michele Boreale	reversible : 16 process : 4	CONCUR,7 FMOODS/FORTE,1 FORTE,2 ICALP,2 MFCS,4 WS-FM,1	false

Εικόνα 3.8 Πίνακας Αποτελεσμάτων



Εικόνα 3.9 Παράθυρο με πληροφορίες λέξεων κλειδιών

Κεφάλαιο 4

Λειτουργία Αλγόριθμου

4.1 Βασική Ιδέα

Το σύστημα έχει ως σκοπό να χρησιμοποιήσει τις εισόδους του χρήστη, οι οποίες είναι τα ονόματα των συγγραφέων ενός επιστημονικού άρθρου, κάποιες λέξεις κλειδιά επιλεγόμενες από τον χρήστη, και τον αριθμό των αποτελεσμάτων που θα εμφανιστούν, και με αυτά να καθορίσει και να παρουσιάσει τους συγγραφείς που είναι πιο κοντά σε αυτές, πράγμα που τους καθιστά και κατάλληλους κριτές για επιστημονικό άρθρο γραμμένο από τους εν λόγω συγγραφείς. Αυτά τα στοιχεία εισόδου θα πρέπει να χρησιμοποιηθούν και σε συνδυασμό με ήδη υπάρχοντα δεδομένα που υπάρχουν σε μια βάση δεδομένων έτσι ώστε να παραχθεί το ορθότερο δυνατό αποτέλεσμα. Για τον σκοπό αυτό χρησιμοποιήθηκαν τα δεδομένα που υπάρχουν στην βάση δεδομένων της ιστοσελίδας DBLP, γιατί θεωρήθηκε ως μια από τις πιο κατάλληλες διαθέσιμες πηγές δεδομένων λόγω του ότι είναι εύκολα διαθέσιμο σε όλους. Ο συσχετισμός ανάμεσα στα δεδομένα εισόδου και στα δεδομένα που βρίσκονται στην βάση δεδομένων θα πρέπει να γίνει από τον αλγόριθμο με τέτοιο τρόπο ώστε να λαμβάνονται υπόψη και το κριτήριο των ‘κοντινότερων συγγραφέων’, αλλά και το κριτήριο των λέξεων κλειδιών. Ως κριτήριο χρησιμοποιήθηκαν διάφορες βιβλιογραφικές πηγές για τους συγγραφείς: πρακτικά συνεδρίων στα οποία έχουν συνεισφέρει, άρθρα περιοδικών στα οποία έχουν γράψει, και ένας συνδυασμός των δύο.

4.1.1 Κριτήριο Καταλληλόλητας

Ο αλγόριθμος που αναπτύχθηκε για το σύστημα, σε πρώτη φάση χρησιμοποιεί τον αλγόριθμο *k*-Nearest-Neighbors[2] για τον καθορισμό των *k* πιο ‘κοντινών’ γειτόνων με

βάση την είσοδο που παίρνει. Ο τρόπος που γίνεται αυτός ο καθορισμός είναι με την χρήση της μετρικής cosine. Ο αλγόριθμος στην δική μας περίπτωση παίρνει ως training data έναν πίνακα στην μορφή sparse matrix, όπου περιέχονται οι πληροφορίες για κάθε συγγραφέα σε ποια άρθρα συνεδρίων/ άρθρα περιοδικών έχει γράψει, και πόσες φορές. Μπορεί να αναπαρασταθεί και ως ένας δισδιάστατος πίνακας όπου η κάθε γραμμή αντιστοιχεί σε έναν συγγραφέα, κάθε στήλη αντιστοιχεί σε ένα συνέδριο/περιοδικό, και κάθε κελί με συντεταγμένες *(x,y)* έχει μέσα τον αριθμό των φορών που ο συγγραφέας *x* συνέβαλε στο *y* (Εικόνα 4.1). Στην συνέχεια, ως είσοδος για τον αλγόριθμο δίνεται ένας ανάλογος πίνακας που αφορά μόνο τους συγγραφείς που μας ενδιαφέρουν (αυτούς για τους οποίους θα βρεθούν οι 'κοντινότεροι γείτονες'). Ο πίνακας αυτός συγκρίνεται με τα training data και βρίσκεται η ομοιότητα με κάθε συγγραφέα χρησιμοποιώντας το κριτήριο μέτρησης cosine similarity, που παίρνει τιμές από -1 μέχρι 1 (-1=διαφορετικά, 1=ίδια). Έτσι επιστρέφονται οι *k* συγγραφείς με την μικρότερη απόσταση από αυτό με βάση το cosine similarity, που στην ουσία είναι μια μαθηματική πράξη που καθορίζει πόσο ίδιοι ή διαφορετικοί είναι πίνακες της μορφής που έχουμε εμείς. Για καλύτερη κατανόηση του όρου cosine similarity, μπορούμε να δούμε πώς θα υπολογιζόταν χωρίς τη χρήση έτοιμης μαθηματικής φόρμουλας για πίνακες όπως αυτούς που αναφέρθηκαν πιο πάνω. Αφού μπουν τα σκορ σε κάθε κελί του πίνακα, η κάθε γραμμή αναπαρίσταται ως ένα σημείο σε μια γραφική παράσταση. Πρακτικά για να γίνει αυτό χρειάζεται γραφική παράσταση με όσες διαστάσεις όσες είναι και οι στήλες του πίνακα μας. Για να υπολογιστεί το cosine similarity μεταξύ δύο τέτοιων σημείων, ενώνουμε το κάθε ένα ξεχωριστά με το σημείο που αρχίζει η γραφική παράσταση (0,0), και υπολογίζουμε το συνημίτονο της γωνιάς που σχηματίζουν οι γραμμές (τιμή μεταξύ -1 και 1). Επιλέχθηκε να χρησιμοποιηθεί αυτός ο αλγόριθμος γιατί είναι από τους καλύτερους και πιο σύνηθες αλγόριθμους που μπορούν να χρησιμοποιηθούν σε συστήματα συστάσεων τέτοιας φύσης όπως αυτού που αναπτύχθηκε, αφού στην ουσία θέλουμε να πάρουμε τους K πιο 'κοντινούς' συγγραφείς με τα δικά μας κριτήρια. Επίσης επιλέχθηκε για το γεγονός ότι, λαμβάνοντας υπόψη τα δεδομένα που έχουμε και την μορφή τους, είναι ο πιο κατάλληλος για να έχει τα καλύτερα αποτελέσματα.

	Conference1	**Conference2**	**Conference3**	**Conference4**
Author1	3	0	1	0
Author2	4	4	0	3
Author3	1	5	2	7
Author4	1	0	0	0

Εικόνα 4.1 Αναπαράσταση sparse matrix σε μορφή δισδιάστατου πίνακα

4.1.2 Απαιτήσεις Αλγόριθμου

i. Να παράγει αποτελέσματα με βάση κάποιο κριτήριο που να καθορίζει συγγραφείς που είναι ‘κοντά’ στους συγγραφείς που παρέχει ο χρήστης ως είσοδο.

ii. Να παράγει αποτελέσματα με βάση τις λέξεις κλειδιά που παρέχει ο χρήστης ως είσοδο.

iii. Να αποκλείονται από τις επιλογές οι συγγραφείς που ήταν συν-συγγραφείς με κάποιον ή κάποιους από τους συγγραφείς που έδωσε ως είσοδο ο χρήστης.

iv. Τα αποτελέσματα που θα παράγονται να είναι έτοιμα σε ένα λογικό χρόνο και να είναι ξεκάθαρα και επεξηγηματικά.

4.1.3 Χειρισμός Δεδομένων από DBLP

Όλα τα δεδομένα που χρησιμοποιούνται από το σύστημα έχουν παρθεί από την βάση δεδομένων της ιστοσελίδας DBLP[1]. Αυτή η διαδικασία έγινε με τα ακόλουθα βήματα:

1. Αρχικά κατέβασα το xml file που είναι διαθέσιμο στην ιστοσελίδα DBLP το οποίο περιέχει μέσα όλες τις καταχωρήσεις στην ιστοσελίδα.

2. Στην συνέχεια, χρησιμοποιώντας script γραμμένο στη Python, εξήγαγα όλα τα στοιχεία τα οποία χρειάζονται για το σύστημα (Ονόματα συγγραφέων, συνέδρια, περιοδικά, τίτλοι καταχωρήσεων), και φυλάχθηκαν σε Dataframes.

3. Τα Dataframes αυτά φυλάχθηκαν σε αρχεία τύπου parquet [20]. Επιλέχθηκε αυτός ο τύπος αρχείου γιατί δίνει την δυνατότητα αποθήκευσης δεδομένων μεγάλου όγκου και χρειάζεται σχετικά μικρή χωρητικότητα από τον δίσκο. Επίσης ο χρόνος που χρειάζεται για να έχουμε πρόσβαση σε δεδομένα αποθηκευμένα σε ένα τέτοιο αρχείο είναι αρκετά μικρός.

4. Τα Dataframes που είναι αποθηκευμένα σε αυτή τη μορφή φορτώνονται στην αρχή του αλγόριθμου από τα parquet files και είναι έτοιμα για να χρησιμοποιηθούν.

4.2 Βασικές Φάσεις Αλγόριθμου

Ο αλγόριθμος που αναπτύχθηκε για το σύστημα μπορεί να χωριστεί στις ακόλουθες φάσεις:

Φάση 1

Εισαγωγή Δεδομένων που χρειάζονται στο πρόγραμμα.

Φάση 2

Επεξεργασία Δεδομένων και δημιουργία πινάκων που θα χρειαστούν στη συνέχεια.

Φάση 3

Επεξεργασία και αντιστοίχιση πινάκων.

Φάση 4

Εφαρμογή αλγόριθμου *k*-Nearest-Neighbors.

Φάση 5

Αφαίρεση συν-συγγραφέων από τα αποτελέσματα.

Φάση 6

Εφαρμογή αλγόριθμου που αφορά τις λέξεις-κλειδιά.

Φάση 7

Ανίχνευση πιθανών συν-συγγραφέων.

Φάση 8

Εύρεση και φύλαξη κοινών άρθρων σε συνέδρια/ περιοδικά.

Φάση 9

Επιστροφή δεδομένων που χρειάζονται.

4.3 Περιγραφή Δομών που αναφέρθηκαν

4.3.1 Dataframe

Ένα Dataframe είναι μια δισδιάστατη και μεταβλητή σε μέγεθος δομή δεδομένων με γραμμές και στήλες, που μοιάζει με έναν πίνακα. Είναι μια από τις πιο συχνά χρησιμοποιούμενες δομές δεδομένων στην ανάλυση και χειρισμό δεδομένων χρησιμοποιώντας τη γλώσσα προγραμματισμού Python. Σε ένα Dataframe, τα δεδομένα οργανώνονται σε μορφή πίνακα όπου κάθε σειρά αντιπροσωπεύει μια μεταβλητή και κάθε στήλη μια άλλη. Τα δεδομένα μπορεί να είναι διαφορετικού είδους, πράγμα που σημαίνει ότι κάθε στήλη μπορεί να περιέχει διαφορετικό τύπο δεδομένων (π.χ. Integer, String κλπ.).

4.3.2 Sparse Matrix

Ένας Sparse Matrix είναι ένας πίνακας στον οποίο αποθηκεύονται στοιχεία που αποτελούνται από τρεις τιμές: τον αριθμό της γραμμής, τον αριθμό της στήλης, και την τιμή που περιέχεται στο αντίστοιχο σημείο. Χρησιμοποιείται σε περιπτώσεις όπου η πλειονότητα των στοιχείων του έχει τιμή ίση με μηδέν, επειδή στον πίνακα αυτό αποθηκεύονται μόνο τα στοιχεία που έχουν μη μηδενική τιμή. Λόγω αυτού, οι πίνακες αυτοί χρησιμοποιούνται συχνά σε αριθμητικούς υπολογισμούς και σε ανάλυση δεδομένων, αφού μειώνουν σημαντικά τις απαιτήσεις αποθήκευσης και υπολογισμού για μεγάλους πίνακες με πολλά μηδενικά στοιχεία. Εκτός της μορφής αναπαράστασης με λίστα συντεταγμένων που προαναφέρθηκε, υπάρχουν και άλλες μορφές όπως συμπιεσμένης αραιής γραμμής, και συμπιεσμένης αραιής στήλης, όπου κάθε μορφή έχει τα δικά της πλεονεκτήματα και μειονεκτήματα.

4.4 Περιγραφή και Ψευδοκώδικας Φάσεων

Ο αλγόριθμος που θα αναπτυχθεί θα πρέπει να καλύπτει τις ανάγκες και τις απαιτήσεις, πράγμα που σημαίνει πως ο αλγόριθμος μπορεί να χωριστεί σε φάσεις, όπως αναφέρθηκαν πιο πάνω, όπου κάθε φάση θα υλοποιεί μια από τις απαιτούμενες λειτουργίες έτσι ώστε στο τέλος να παράγει το επιθυμητό αποτέλεσμα. Ακολουθεί ο ψευδοκώδικας, και εκτενής επεξήγηση για κάθε φάση.

Φάση 1

Ψευδοκώδικας

```
1. Εισαγωγή Dataframe df1 με πληροφορίες για άρθρα Συνεδρίων/περιοδικών
2. Εισαγωγή Dataframe df2 με πληροφορίες για τίτλους
3. Για κάθε είσοδο του χρήστη:
4.      Αν είναι όνομα συγγραφέα:
5.          Πρόσθεσε το όνομα στην λίστα listKeys
6.      Αν είναι λέξη κλειδί:
7.          Πρόσθεσε την λέξη κλειδί στην λίστα listNames
```

Επεξήγηση

Αρχικά, ο αλγόριθμος επεξεργάζεται τα στοιχεία που δόθηκαν ως είσοδο από τον χρήστη (γραμμές 3-7) και εισάγει στο πρόγραμμα τα δεδομένα από το DBLP με την μορφή pandas Dataframes έτσι ώστε να είναι δυνατή η εύκολη και γρήγορη επεξεργασία τους (γραμμές 1-2). Τα entries στο df1 που εισάγεται είναι της μορφής: όνομα συγγραφέα, όνομα συνέδριου ή/και περιοδικού, αριθμός συμβολών του συγκεκριμένου συγγραφέα στο συγκεκριμένο συνέδριο/περιοδικό (Εικόνα 4.4.1), και στο df2 είναι της μορφής όνομα συγγραφέα, τίτλος καταχώρησης του συγκεκριμένου συγγραφέα στην βάση δεδομένων της DBLP (Εικόνα 4.4.2).

	author	booktitle	ratings
Anna Philippou	ATAED@Petri	Nets/ACSD	1

Εικόνα 4.4.1 Παράδειγμα καταχώρησης στο Dataframe df1

	author	title
"Johann" Sebastian Rudolph		Presburger Concept Cardinality Constraints in ...

Εικόνα 4.4.2 Παράδειγμα καταχώρησης στο Dataframe df2

Φάση 2

Ψευδοκώδικας

8. Δημιουργία Dataframe dfnames με όλα τα ονόματα συγγραφέων χωρίς διπλότυπα
9. Δημιουργία Dataframe dfconf με όλα τα ονόματα συνέδριων/περιοδικών χωρίς διπλότυπα
10. Δημιουργία και πρόσθεση νέας στήλης με τα σκορ στο Dataframe dfconf

11. Συνάρτηση δημιουργίας sparse matrix:
    a. Δημιουργία μεταβλητής rows για τις γραμμές που αντιστοιχούν στους συγγραφείς
    b. Δημιουργία μεταβλητής cols για τις στήλες που αντιστοιχούν στα συνέδρια
    c. Δημιουργία μεταβλητής rating για τις τιμές που αντιστοιχούν στα σκορ
    d. Δημιουργία και επιστροφή του sparse matrix με την συνάρτηση csr_matrix(rating,(rows, cols))
12. Δημιουργία sparse matrix με τις στήλες όνομα συγγραφέα, όνομα συνέδριου, σκορ από το πρώτο Dataframe df1 που εισάχθηκε, με την χρήση της συνάρτησης

13. Δημιουργία καινούργιου κενού Dataframe dfcomb
14. Για κάθε όνομα συγγραφέα μέσα στην λίστα listNames με τα ονόματα εισόδου:
    e. Προσθέτετε στο dfcomb το κομμάτι από το df1 που αφορά τον συγγραφέα

Επεξήγηση

Με βάση τις εισόδους του χρήστη και τα Dataframes που αναφέρθηκαν στην προηγούμενη φάση, δημιουργούνται τρία νέα Dataframes και ένας sparse matrix, που θα χρησιμοποιηθούν για να καθοριστούν οι ‘κοντινότεροι’ συγγραφείς σε αυτούς που δόθηκαν ως είσοδο από τον χρήστη. Στο πρώτο Dataframe dfnames περιέχονται όλα τα ονόματα συγγραφέων που υπάρχουν στην βάση δεδομένων της DBLP χωρίς διπλότυπα. Στο δεύτερο Dataframe dfconf

περιέχονται όλοι οι τίτλοι των συνεδρίων/περιοδικών που υπάρχουν μέσα στην βάση δεδομένων της DBLP χωρίς διπλότυπα, και ένα σκορ για κάθε συνέδριο το οποίο στην αρχή είναι για όλα ίσο με μηδέν (Εικόνα 4.4.3). Μέσα στον sparse matrix περιέχονται σημεία της μορφής (x,y) και ένα αντίστοιχο σκορ s, τα οποία αντιστοιχούν στο ακόλουθο: 'ο x συγγραφέας έχει συμβάλει s φορές στο συνέδριο y' (Εικόνα 4.4.4). Ο λόγος που χρησιμοποιείται sparse matrix είναι λόγω θέματος χωρητικότητας, αφού αυτός ο τρόπος φύλαξης αυτών των δεδομένων απαιτεί σημαντικά λιγότερο χώρο από άλλες επιλογές, και λόγω της ύπαρξης μεγάλου όγκου δεδομένων οποιαδήποτε μείωση του χώρου που χρειάζεται είναι σημαντική. Σε αυτό το σημείο δημιουργείται και ένα τρίτο Dataframe, στο οποίο φυλάγονται όλα τα entries από την βάση δεδομένων της DBLP που αφορούν τους συγγραφείς που έδωσε ο χρήστης ως είσοδο. Τα δεδομένα αυτά φυλάγονται σε μορφή όνομα συγγραφέα, όνομα συνεδρίου, αριθμός συμβολών του συγκεκριμένου συγγραφέα στο συγκεκριμένο συνέδριο/περιοδικό (Εικόνα 4.4.1). Σημείωση: Στην εκδοχή του αλγόριθμου που λαμβάνει υπόψη και συνέδρια και περιοδικά, η πιο πάνω διαδικασία γίνεται δύο φορές, μια για τα συνέδρια και μια για τα περιοδικά.

booktitle	ratings
#MSM	0

Εικόνα 4.4.3 Παράδειγμα καταχώρησης στο Dataframe dfconf

```
(0, 3280)     1
(1, 3940)     1
(2, 6089)     1
(3, 2197)     1
(3, 6088)     1
```

Εικόνα 4.4.4 Παράδειγμα καταχωρήσεων στον sparse matrix

Φάση 3

Ψευδοκώδικας

```
15.    Για κάθε συνέδριο/περιοδικό στο dfconf:
16.        Για κάθε entry στο Dataframe  dfcomb:
17.            Αν το συνέδριο είναι το ίδιο με το
   αντίστοιχο στο entry:
18.                Πρόσθεση του σκορ του συνέδριου στην
   ανάλογη στήλη του Dataframe από το df1.
```

Επεξήγηση

Στο σημείο αυτό γίνεται η αντιστοίχιση μεταξύ του πρώτου Dataframe dfconf και του δεύτερου Dataframe dfcomb, όπου στο dfcomf μπαίνουν τα σκορ τα οποία έχουν οι συγγραφείς εισόδου στα αντίστοιχα συνέδρια/περιοδικά. Συγκεκριμένα τα δεδομένα είναι της μορφής συνέδριο/περιοδικό, σκορ στο συγκεκριμένο συνέδριο (Εικόνα 4.4.3). Σημείωση: Στην εκδοχή του αλγόριθμου που λαμβάνει υπόψη και συνέδρια και περιοδικά, η πιο πάνω διαδικασία γίνεται δύο φορές, μια για τα συνέδρια και μια για τα περιοδικά.

Φάση 4

Ψευδοκώδικας

```
19.    Δημιουργία εισόδου για τον αλγόριθμο k Nearest
       Neighbors, που είναι η στήλη με τα σκορ από το Dataframe
       dfcomf

20.    Δημιουργία μοντέλου με καθορισμό χρήσης 'cosine'
       μετρικού

21.    Χρήση του sparse matrix που δημιουργήθηκε νωρίτερα
       ως training data, για προσαρμογή του αλγόριθμου

22.    Εκτέλεση του αλγόριθμου με την είσοδο που
       δημιουργήθηκε πριν, για εύρεση των 200 κοντινότερων
       γειτόνων

23.    Για κάθε αποτέλεσμα του αλγόριθμου:
24.    Αν το όνομα του συγγραφέα δεν υπάρχει μέσα στην
       λίστα αποτελεσμάτων:
25.    Πρόσθεση του ονόματος στην λίστα αποτελεσμάτων
       resList
```

Επεξήγηση

Στη φάση αυτή εφαρμόζεται ο αλγόριθμος που θα καθορίζει τους 'κοντινότερους' συγγραφείς με βάση τα σκορ κάθε συνεδρίου/περιοδικού. Ο αλγόριθμος που επιλέχτηκε για

να χρησιμοποιηθεί είναι ο αλγόριθμος Μηχανικής Μάθησης k Nearest Neighbors, που ήταν διαθέσιμος σε βιβλιοθήκη της Python, ο οποίος με βάση τα κριτήρια που δόθηκαν θα καθορίσει του Κ ‘κοντινότερους’ συγγραφείς χρησιμοποιώντας το μετρικό κριτήριο cosine. Για να γίνει fit χρησιμοποιείται ο sparse matrix που αναφέρθηκε πιο πάνω, και ως είσοδος για να δοθούν αποτελέσματα δίνεται η στήλη με τα σκορ από το dfcomf που αναφέρθηκε πιο πάνω. Όλα τα ονόματα συγγραφέων που δίνονται ως αποτέλεσμα από τον αλγόριθμο φυλάγονται σε μία λίστα αποτελεσμάτων. Σημείωση: Στην εκδοχή του αλγόριθμου που λαμβάνει υπόψη και συνέδρια και περιοδικά, η πιο πάνω διαδικασία γίνεται δύο φορές, μια για τα συνέδρια και μια για τα περιοδικά.

Φάση 5

Ψευδοκώδικας

26. Συνάρτηση εύρεσης συν-συγγραφέων(συγγραφέας):
    i. Καθορισμός URL που θα χρησιμοποιηθούν για xml requests
    ii. Εύρεση ατομικού κωδικού συγγραφέα με το πρώτο URL
    iii. Δημιουργία λίστα συν-συγγραφέων fcoList
    iv. Εύρεση συν-συγγραφέων του συγγραφέα με την χρήση του ατομικού του κωδικού και του δεύτερου URL
    v. Πρόσθεση ονομάτων συν-συγγραφέων στην λίστα συν-συγγραφέων
    vi. Επιστροφή λίστας συν-συγγραφέων fcoList

27. Δημιουργία λίστας συν-συγγραφέων coList
28. Για κάθε όνομα συγγραφέα στην λίστα εισόδων lisNames:
    i. Εφαρμογή συνάρτησης εύρεσης συν-συγγραφέων και πρόσθεση αποτελεσμάτων στην λίστα συν-συγγραφέων coList
29. Διαγραφή διπλότυπων από την λίστα coList

30. Για κάθε όνομα συγγραφέα στην λίστα αποτελεσμάτων resList:
    i. Αν το όνομα υπάρχει στη λίστα συν-συγγραφέων coList:
        1. Αφαίρεση του ονόματος από την λίστα αποτελεσμάτων resList

Επεξήγηση

Μια σημαντική απαίτηση του αλγόριθμου είναι να αποκλείει από τα αποτελέσματα συγγραφείς οι οποίοι υπήρξαν συν-συγγραφείς με του συγγραφείς που δόθηκαν ως είσοδο από τον χρήστη. Αυτός ο έλεγχος πραγματοποιείται σε αυτό το σημείο. Για κάθε έναν

συγγραφέα που έδωσε ο χρήστης ως είσοδο, παίρνουμε όλους τους συν-συγγραφείς του με την χρήση xml request που παρέχει η DBLP, και τους προσθέτουμε στην λίστα coList. Στο τέλος αυτής της διαδικασίας έχουμε μια λίστα που περιέχει όλους τους συν-συγγραφείς όλων των συγγραφέων που δόθηκαν για είσοδο. Στην συνέχεια γίνεται σύγκριση μεταξύ αυτής της λίστας coList και με τα αποτελέσματα που έδωσε πιο πριν ο αλγόριθμος k Nearest Neighbors που βρίσκονται στη λίστα resList, και όποιος συγγραφέας είναι κοινός μεταξύ των δύο λιστών, αφαιρείται από την λίστα των αποτελεσμάτων resList. Σε κάποιες ειδικές περιπτώσεις μπορεί κάποιος συν-συγγραφέας να μην ανιχνευθεί λόγω διαφοράς μεταξύ των ονομάτων που παρέχει η DBLP, αλλά αυτό το θέμα αντιμετωπίζεται στην συνέχεια.

Φάση 6

Ψευδοκώδικας

31. Συνάρτηση για λέξεις κλειδιά(συγγραφέας):
    i. Εύρεση όλων των τίτλων των καταχωρήσεων του συγγραφέα από Dataframe df2 που εισάχθηκε στην αρχή
    ii. Μετατροπή και φύλαξη των τίτλων σε μια λίστα tl1
    iii. Αφαίρεση τίτλων που είναι άδειοι
    iv. Δημιουργία λίστας για φύλαξη των αποτελεσμάτων tlres
    v. Για κάθε λέξη κλειδί στην λίστα listKeys που έδωσε ο χρήστης:
        1. Δημιουργία της λίστας tl2 για φύλαξη των τίτλων που περιέχουν την λέξη κλειδί
        2. Για κάθε τίτλο μέσα στην αρχική λίστα τίτλων tl1:
            a. Αν η λέξη κλειδί περιέχεται μέσα στον τίτλο και δεν ξαναβρέθηκε πιο πριν:
                i. Πρόσθεση ενός πόντου στο σκορ
                ii. Πρόσθεση του τίτλου στην νέα λίστα tl2
                iii. Φύλαξη της λέξης κλειδιού, του αντίστοιχου σκορ και των αντίστοιχων τίτλων σε λίστα αποτελεσμάτων tlres
    vi. Επιστροφή λίστας αποτελεσμάτων tlres

32. Για κάθε όνομα συγγραφέα στην λίστα listNames:
    i. Εφαρμογή της συνάρτησης για λέξεις κλειδιά και φύλαξη του αποτελέσματος

33. Για κάθε ένα από τα αποτελέσματα της συνάρτησης:

i. Υπολογισμός και φύλαξη του συνολικού σκορ όλων των λέξεων κλειδιών από το αντίστοιχο αποτέλεσμα μαζί

34. Ταξινόμηση όλων των παράλληλων πινάκων αποτελεσμάτων σε φθίνουσα σειρά με βάση το σκορ των λέξεων κλειδιών, και συγκρατούνται τα πρώτα 100

Επεξήγηση

Η φάση αυτή είναι το κομμάτι του αλγόριθμου που αφορά τις λέξεις κλειδιά που δόθηκαν από τον χρήστη ως είσοδος. Για κάθε συγγραφέα που έμεινε στη λίστα των αποτελεσμάτων του αλγόριθμου k Nearest Neighbors μετά το προηγούμενο βήμα, βρίσκουμε όλους τους τίτλους των καταχωρήσεων του συγκεκριμένου από το df2 χωρίς διπλότυπα. Ανάλογα με το πόσοι τίτλοι κάποιου συγγραφέα περιέχουν κάποια από τις λέξεις κλειδιά, προστίθενται οι ανάλογοι πόντοι στο σκορ του συγκεκριμένου συγγραφέα. Για παράδειγμα αν σε 5 από τους τίτλους των καταχωρήσεων του συγγραφέα χ περιέχεται η λέξη κλειδί 1 και σε 4 η λέξη κλειδί 2, φυλάγονται οι αριθμοί 5 και 4 ως σκορ της αντίστοιχης λέξης κλειδιού, αλλά και ο αριθμός 9 ως το συνολικό σκορ όλων των λέξεων κλειδιών. Αυτά τα στοιχεία υπολογίζονται και φυλάγονται για κάθε συγγραφέα στη λίστα αποτελεσμάτων. Φυλάγονται επίσης οι τίτλοι που περιέχουν τις αντίστοιχες λέξεις κλειδιά. Ακολούθως, τα ονόματα των συγγραφέων ταξινομούνται σε φθίνουσα σειρά με βάση το συνολικό σκορ καθενός για τις λέξεις κλειδιά, και δημιουργούνται παράλληλοι πίνακες για να φυλάγονται όλα τα δεδομένα που μας ενδιαφέρουν στη σωστή σειρά. Από τα αποτελέσματα που έχουμε, κρατούμε τους πρώτους 100 συγγραφείς με βάση το σκορ τους.

Φάση 7

Ψευδοκώδικας

35. Δημιουργία πίνακα pco όπου κάθε θέση αντιστοιχεί με ένα συγγραφέα, και αρχικοποίηση όλων των τιμών με False
36. Για κάθε όνομα συγγραφέα στην λίστα resList:
    i. Για κάθε συν-συγγραφέα από την λίστα συν-συγγραφέων coList:
        1. Αν το όνομα συν-συγγραφέα περιέχεται στο όνομα συγγραφέα ή το αντίθετο:
            a. Η αντίστοιχη θέση στον πίνακα pco παίρνει τιμή True

Επεξήγηση

Στο παρών βήμα γίνεται η διαδικασία εύρεσης πιθανού συν-συγγραφέα, όπως αναφέρθηκε στη φάση 5. Ελέγχεται ξανά η λίστα των αποτελεσμάτων resList που έχουμε με την λίστα των συν-συγγραφέων coList. Στην περίπτωση που κάποιο όνομα από την μια λίστα περιέχεται ακριβώς το ίδιο στην άλλη λίστα με την διαφορά κάποιον γραμμάτων ή αριθμών, θέτεται η τιμή σε ένα παράλληλο πίνακα true, που σημαίνει πως όταν τα αποτελέσματα θα παρουσιαστούν στον χρήστη, θα φαίνεται μια επισήμανση πως ο συγκεκριμένος συγγραφέας είναι πιθανός συν-συγγραφέας αυτών που δόθηκαν ως είσοδο. Αυτό γίνεται γιατί σε κάποιες περιπτώσεις υπάρχουν συνωνυμίες και χρησιμοποιούνται αριθμοί στα ονόματα, αλλά και για το λόγο ότι μπορεί να υπάρχουν μικρές διαφορές στο όνομα ίδιου συγγραφέα από τις πηγές πληροφοριών που χρησιμοποιούνται από το DBLP.

Φάση 8

Ψευδοκώδικας

37. Για κάθε όνομα συγγραφέα στην λίστα namesList:
    1. Εύρεση όλων των συνεδρίων που συμμετείχε και φύλαξη τους σε λίστα

38. Για κάθε όνομα συγγραφέα στην λίστα αποτελεσμάτων resList:
    i. Εύρεση όλων των συνεδρίων που συμμετείχε και πρόσθεση του σε λίστα συνεδρίων

39. Για κάθε συνέδριο στην λίστα συνεδρίων:
40. Αν το συνέδριο δεν υπάρχει και στις δύο πιο πάνω λίστες
    i. Πρόσθεση ονόματος συνεδρίου σε λίστα για διαγραφή

41. Διαγραφή όλων των συνεδρίων που υπάρχουν στην λίστα συνεδρίων και στην λίστα για διαγραφή

Επεξήγηση

Ακολούθως, ο στόχος αυτού του βήματος είναι η εύρεση των συνεδρίων/περιοδικών όπου οι συγγραφείς στην λίστα αποτελεσμάτων resList έχουν κοινά με αυτούς που δόθηκαν ως είσοδο, έτσι ώστε να μπορούν αυτά τα δεδομένα να παρουσιαστούν στον χρήστη για καλύτερη του πληροφόρηση. Αρχικά δημιουργείται μια λίστα με όλα τα συνέδρια/περιοδικά που συμμετείχαν οι συγγραφείς που δόθηκαν ως είσοδος, χρησιμοποιώντας τα Dataframes που υπήρχαν ήδη από τα προηγούμενα βήματα. Στην συνέχεια, για κάθε συγγραφέα στην λίστα αποτελεσμάτων resList, βρίσκουμε και για αυτούς όλα τα συνέδρια/περιοδικά στα οποία συμμετείχαν, και από αυτή την λίστα αφαιρούνται τα συνέδρια/περιοδικά που δεν υπάρχουν στην λίστα των συγγραφέων που δόθηκαν ως είσοδος, τα μη κοινά συνέδρια/περιοδικά μεταξύ τους δηλαδή.

Φάση 9

Ψευδοκώδικας

```
42. Πρόσθεση σε γενική λίστα αποτελεσμάτων allRes την λίστα με
τα ονόματα των συγγραφέων που προήλθαν από τον αλγόριθμο k
Nearest Neighbors (resList), και τις παράλληλες λίστες με τις
λέξεις κλειδιά και τα αντίστοιχα σκορ και τίτλους στους
οποίους περιέχονται, τα συνέδρια που συμμετείχε ο κάθε
συγγραφέας, και η λίστα που δείχνει ποιοι συγγραφείς είναι
πιθανοί συν-συγγραφείς(pco)

43. Επιστροφή της  γενικής λίστας αποτελεσμάτων allRes με την
μορφή JSON
```

Επεξήγηση

Σε αυτό το σημείο είναι το τέλος του αλγορίθμου, και στέλνονται 5 παράλληλες λίστες σε μορφή JSON για να τύχουν επεξεργασίας από την JavaScript. Η πρώτη λίστα περιέχει τα ονόματα των 100 πρώτων συγγραφέων με βάση τα σκορ τους από τις λέξεις κλειδιά, η δεύτερη περιέχει το αντίστοιχο σκορ από τις λέξεις κλειδιά και η τρίτη λίστα τα αντίστοιχα σκορ των λέξεων κλειδιών ξεχωριστά για κάθε μια αλλά και τους αντίστοιχους τίτλους στους οποίους περιέχονται. Η τέταρτη λίστα περιέχει τα κοινά συνέδρια του κάθε συγγραφέα με τους συγγραφείς που δόθηκαν ως είσοδος, και η πέμπτη λίστα περιέχει την πληροφορία αν ο αντίστοιχος συγγραφέας είναι πιθανός συν-συγγραφέας.

Κεφάλαιο 5

Αξιολόγηση

5.1 Δοκιμές Αλγόριθμου

Έτρεξα τον αλγόριθμο με τις ακόλουθες εισόδους για k=100, k=200 και k=300, και για όλες τις εκδοχές του. Στην Εικόνα 5.1 φαίνεται η γραφική παράσταση του χρόνου εκτέλεσης του αλγόριθμου σε αυτές τις περιπτώσεις.
Πιο κάτω φαίνονται όλες οι εισόδοι με τες οποίες έτρεξα τον αλγόριθμο. Κάθε είσοδος αποτελείται από ονόματα συγγραφέων και λέξεις κλειδιά.

Anna Philippou, David Walker - confluece, determinacy, process calculus

Anna Philippou, Kyriaki Psara - reversibility, Petri nets

Anna Philippou, Evangelia Vanezi - privacy, formal methodsd

Dimitrios Kouzapas, Nobuko Yoshida - session types

Maciej Koutny, Lukasz Mikulski - multi-agent systems

Jetty Kleijn, Maciej Koutny - membrane systems

James Hoey, Irek Ulidowski - Reversibility, Concurrent Programs

Hernán C. Melgratti, Claudio Antares Mezzina - Reversing, nets

Irek Ulidowski, Iain Phillips - reversibility, event structures

Ivan Lanese, Neil Sayers, Emilio Tuosto - Design-by-Contract, multiparty session types

Franco Barbanera, Ivan Lanese - Choreographic Languages

Ivan Lanese, Adrián Palacios, Germán Vidal - debugging, Erlang

Luca Aceto, Adrian Francalanza - run-time verification, first-order properties

Luca Aceto, Antonis Achilleos - axiomatization, parallel composition

Adrian Francalanza, Anna Ingólfsdóttir - benchmarking, runtime verification

Mohammad Reza Mousavi, Ana Cavalcanti - Autonomous Systems, formal methods

Jan Friso Groote, Mohammad Reza Mousavi - compositional learning, automata

Andrea Margheri, Vladimiro Sassone - access control, privacy

Gethin Norman, David Parker - model checking, autonomous systems

Ornela Dardha, Jorge A. Pérez - type systems, deadlock freedom

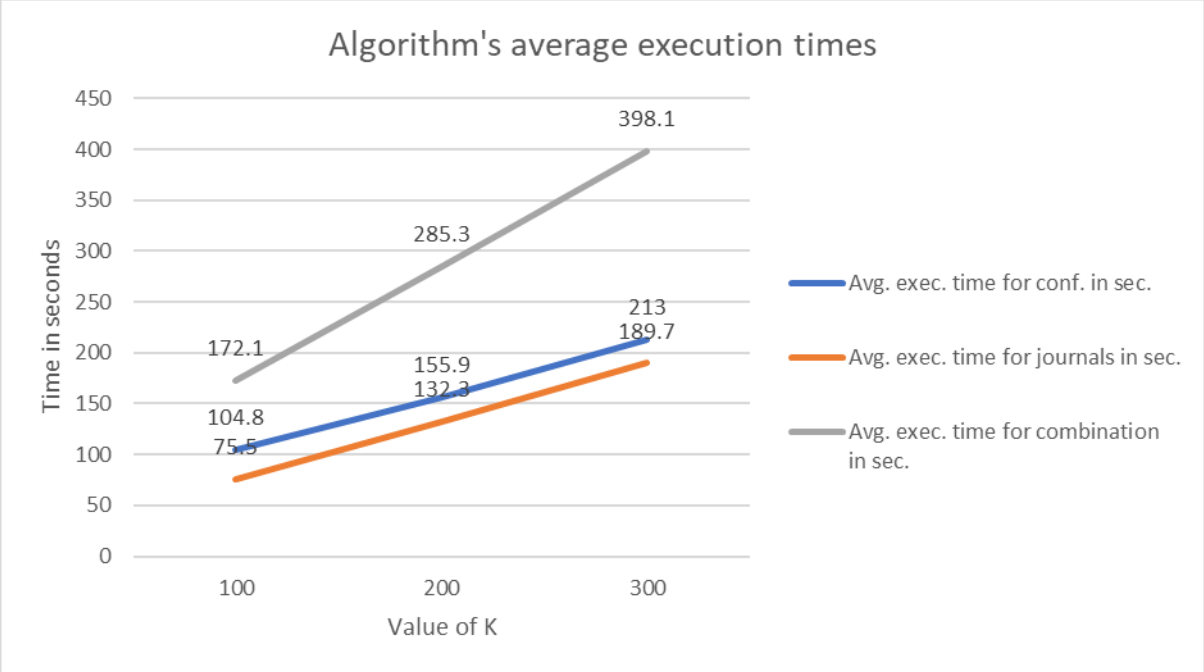
Tobias Runge, Alex Potanin, Ina Schaefer - Correct-by-Construction, Programming

Laura Bocchi, Ivan Lanese, Claudio Antares Mezzina, Shoji Yuen - reversibility, time

Kirstin Peters, Uwe Nestmann, Christoph Wagner - fault-tolerant, multi-party session types

Emmanuel Paviot-Adet, Denis Poitrenaud, Etienne Renault, Yann Thierry-Mieg - LTL, stutter invariance

Florian Renkin, Philipp Schlehuber-Caissier, Alexandre Duret-Lutz, Adrien Pommellet - Mealy machines, reductions



Εικόνα 5.1 Γραφική παράσταση χρόνου εκτέλεσης του αλγόριθμου

Παρατηρήσεις

Όπως φαίνεται από την γραφική παράσταση στην Εικόνα 5.1, ο χρόνος εκτέλεσης αυξανόταν σε όλες τις εκδοχές του αλγόριθμου όσο αυξανόταν το *k*. Μικρότερο χρόνο εκτέλεσης είχε η

εκδοχή με τα περιοδικά, με μέσους όρους 75.5s, 132.3s και 189.7s για *k* ίσο με 100, 200 και 300 αντίστοιχα. Ακολουθεί η εκδοχή του αλγόριθμου με τα συνέδρια όπου για *k*=100 ο μέσος όρος εκτέλεσης ήταν 104.8s, για *k*=200 ήταν 155.9s και για *k*=300 ο μέσος όρος εκτέλεσης ήταν 213s. Τέλος, η εκδοχή του αλγόριθμου που είχε τους μεγαλύτερους χρόνους εκτέλεσης ήταν αυτή που λάμβανε υπόψη και τα συνέδρια και τα περιοδικά, όπως ήταν αναμενόμενο. Ο μέσος χρόνος εκτέλεσης για *k*=100 ήταν 172.1s, για *k*=200 ήταν 285.3s και για *k*=300 ήταν 398.1s.

5.2 Συμπεράσματα με Βάση τα Αποτελέσματα

Χρόνος Εκτέλεσης

Αρχικά, συγκρίνοντας τον χρόνο εκτέλεσης του αλγόριθμου στις διάφορες εκδοχές του, βλέπουμε πως στην εκδοχή που λαμβάνονται υπόψη τα κοινά συνέδρια και σε αυτή που λαμβάνονται υπόψη τα κοινά άρθρα σε περιοδικά ο χρόνος εκτέλεσης είναι αρκετά κοντά για παρόμοια *k*, με την πρώτη δεύτερη να είναι λίγο πιο γρήγορη από την πρώτη αντίστοιχα. Βλέπουμε, όμως, ότι με αντίστοιχα *k* ο αλγόριθμος που λαμβάνει υπόψη και τα δύο κριτήρια είναι πιο αργός από τους άλλους, κάτι το οποίο ήταν αναμενόμενο. Όσο αφορά το *k*, παρατηρούμε για κάθε αλγόριθμο πως όσο μεγαλώνει το *k*, μεγαλώνει και ο χρόνος εκτέλεσης του αντίστοιχου αλγόριθμου.

Παρατηρήσεις και Ποιότητα Αποτελεσμάτων

Αρχικά, αναφέρω πως τα συμπεράσματα για την ποιότητα των αποτελεσμάτων βγήκαν μετά από συζητήσεις με την Επιβλέπουσα Καθηγήτρια της Ατομικής Διπλωματικής μου Εργασίας κυρία Άννα Φιλίππου. Λαμβάνοντας υπόψη τα *k*, βλέπουμε ότι τα αποτελέσματα είναι πολύ καλύτερα με *k*=200 αντί με *k*=100, αφού μεγαλώνει η εμβέλεια του αλγόριθμου και λαμβάνονται υπόψη ονόματα που στην περίπτωση με *k*=100 δεν θα λαμβάνονταν υπόψη. Συγκρίνοντας τα αποτελέσματα όταν το *k*=200 και *k*=300, βλέπουμε ξανά πως κάποια ονόματα διαφέρουν και κάποια είναι κοινά μεταξύ των δύο, όμως η αλλαγή στην ποιότητα των αποτελεσμάτων δεν είναι τόσο μεγάλη όσο η διαφορά μεταξύ των *k*=100 και *k*=200. Ακολούθως, στην σύγκριση ανάμεσα στις εκδοχές του αλγόριθμου, βλέπουμε πως η ποιότητα των αποτελεσμάτων του αλγόριθμου που λαμβάνει υπόψη τα κοινά συνέδρια είναι πολύ μεγαλύτερη από αυτή του αλγόριθμου που λαμβάνει υπόψη τα κοινά άρθρα σε περιοδικά. Θεωρούμε πως αυτό οφείλεται στο γεγονός ότι η θεματολογία των άρθρων είναι πολύ μεγαλύτερη από αυτή των συνεδρίων, δίνοντας έτσι αποτελέσματα από μεγαλύτερη εμβέλεια και όχι τόσο εύστοχα. Αυτό φαίνεται και αν συγκρίνουμε τον αλγόριθμο των συνεδρίων με αυτό που λαμβάνει υπόψη και τα συνέδρια και τα κοινά άρθρα σε περιοδικά,

όπου παρατηρούμε μια μεγάλη ομοιότητα στα αποτελέσματα. Συγκεκριμένα, από τις πιο πάνω δοκιμές προέκυψε το ακόλουθο στατιστικό: τα αποτελέσματα του αλγόριθμου που λαμβάνει υπόψη και τα συνέδρια και τα περιοδικά είναι κατά μέσο όρο 91% όμοια με τα αποτελέσματα του αλγόριθμου με τα συνέδρια, ενώ είναι κατά μέσο όρο μόνο 10% όμοια με αυτόν που λαμβάνει υπόψη τα περιοδικά. Αυτό συμβαίνει επειδή, όσα ονόματα λαμβάνονται υπόψη στον πρώτο αλγόριθμο λαμβάνονται και στον άλλο με επιπρόσθετα ονόματα που προέρχονται από κοινά άρθρα σε περιοδικά, και επειδή στις πλείστες περιπτώσεις τα αποτελέσματα που προέρχονται από τα κοινά συνέδρια είναι καλύτερα, προτιμώνται παρά τα άλλα. Μια άλλη σημαντική παρατήρηση είναι ότι τα αποτελέσματα του αλγόριθμου εξαρτώνται άμεσα από τις λέξεις-κλειδιά που δίνει ο χρήστης. Αν δοθούν ακατάλληλες λέξεις-κλειδιά τότε και τα αποτελέσματα δεν θα είναι καλά, και η ποιότητα των λέξεων-κλειδιών φαίνεται από τα αποτελέσματα. Επίσης πρέπει να δίνεται προσοχή στα ονόματα που δίνονται ως είσοδοι γιατί μπορεί να υπάρχουν συνωνυμίες και θα πρέπει να γράφονται ακριβώς όπως στο DBLP.

Γενικά Συμπεράσματα

Από τα αποτελέσματα και τις συγκρίσεις που έγιναν πιο πάνω, καταλήξαμε σε κάποια συμπεράσματα όσο αφορά ποιος συνδυασμός εκδοχής του αλγόριθμου και του k είναι πιο κατάλληλος για να χρησιμοποιηθεί από το σύστημα. Αρχίζοντας από την εκδοχή του αλγόριθμου, η πιο κατάλληλη είναι αυτή που λαμβάνει υπόψη τα συνέδρια, αφού τα αποτελέσματα που δίνει είναι πολύ καλύτερα από την εκδοχή που λαμβάνει υπόψη τα κοινά άρθρα σε περιοδικά. Επίσης, ο χρόνος εκτέλεσης είναι αρκετά πιο μικρός σε σχέση με τον αλγόριθμο που λαμβάνει υπόψη και τα κοινά συνέδρια και τα κοινά άρθρα σε περιοδικά, και η ποιότητα των αποτελεσμάτων είναι σχεδόν η ίδια. Όσο αφορά τα k, θεωρήθηκε ως πιο κατάλληλος αριθμός το 200, λόγω του ότι τα αποτελέσματα που παρέχει είναι πολύ καλύτερα από όταν k=100. Επίσης, όταν ο αλγόριθμος χρησιμοποιεί k=200, ο χρόνος εκτέλεσης του είναι αρκετά πιο γρήγορος από αυτόν που χρησιμοποιεί k=300, και οι μικρές βελτιώσεις που υπάρχουν με k=300 δεν είναι ουσιαστικές.

Κεφάλαιο 6

Συμπεράσματα

6.1 Γενική Ανασκόπηση

Στόχος της Ατομικής Διπλωματικής μου Εργασίας ήταν η ανάπτυξη ενός συστήματος συστάσεων (recommender system), το οποίο θα έκανε συστάσεις για κριτές για επιστημονικά άρθρα. Το σύστημα θα έπρεπε να χρησιμοποιεί εισόδους όπως ονόματα συγγραφέων ενός άρθρου, και με βάση δεδομένα από τη βάση δεδομένων της ιστοσελίδας DBLP, και εφαρμόζοντας τον αλγόριθμο μηχανικής μάθησης (machine learning) *k*-Nearest-Neighbors να επιστρέφει ονόματα πιθανών κριτών για το συγκεκριμένο άρθρο. Αφού έγιναν κάποιες αρχικές δοκιμές και εξέταση αποτελεσμάτων, κρίθηκε αναγκαίο το σύστημα να έχει την δυνατότητα να δέχεται ως είσοδο και λέξεις-κλειδιά, και αφού τρέξει ο πρώτος αλγόριθμος να γίνεται post-processing των αποτελεσμάτων που έδωσε χρησιμοποιώντας αλγόριθμο για τις λέξεις-κλειδιά. Ένα τέτοιο σύστημα όπως αυτό που περιεγράφηκε θα έκανε τη διαδικασία επιλογής κριτών για επιστημονικά άρθρα αυτοματοποιημένα, πράγμα που σημαίνει πως μειώνεται ο χρόνος και ο κόπος που θα χρειάζονταν αν αυτή η διαδικασία γινόταν χειροκίνητα από κάποιο άτομο. Επίσης, το σύστημα αυτό θα είναι αξιόπιστο, θα παρέχει, δηλαδή, συστάσεις που, με βάση τα δεδομένα εισόδου που δόθηκαν, θα είναι ορθές.

Για να αναπτύξω το σύστημα συστάσεων, έπρεπε πρώτα να εξοικειωθώ με διάφορες τεχνολογίες και έννοιες με τις οποίες δεν ήμουν από πριν. Η εκμάθηση των γλωσσών προγραμματισμού HTML, CSS, JavaScript, και PHP είναι κάποια παραδείγματα γλωσσών που, πριν από την Διπλωματική μου Εργασία, είχα δουλέψει μαζί τους ελάχιστα, και χρειάστηκε να τις μάθω έτσι ώστε να μπορέσω να σχεδιάσω την ιστοσελίδα του συστήματος αλλά και να εκτελούνται οι λειτουργίες του. Μία από τις κύριες δυσκολίες που αντιμετώπισα σε αυτό τον τομέα ήταν η επικοινωνία μεταξύ της HTML, της JavaScript, της PHP, και του αλγόριθμου ο οποίος είναι γραμμένος στην γλώσσα προγραμματισμού Python. Δηλαδή, η

δυσκολία ήταν στο πώς τα στοιχεία εισόδου που δίνονταν από τον χρήστη θα περάσουν από όλα τα πιο πάνω βήματα και στο τέλος θα σταλούν στον αλγόριθμο για να εκτελεστεί με αυτά. Η λύση αυτού του προβλήματος ήταν, παίρνοντας τις εισόδους στην HTML, στέλνονται στην JavaScript, όπου μετά από κάποια επεξεργασία στέλνονται στην PHP, και από εκεί, χρησιμοποιώντας εντολές που υπάρχουν στην PHP, καλείται ο αλγόριθμος γραμμένος σε Python με ορίσματα τα στοιχεία εισόδου. Κάποιες άλλες έννοιες που χρειάστηκε να μάθω είχαν να κάνουν με αλγόριθμους Μηχανικής Μάθησης, και συγκεκριμένα τον αλγόριθμο *k*-Nearest-Neighbors ο οποίος χρησιμοποιήθηκε στον αλγόριθμο του συστήματος. Μια δυσκολία που αντιμετώπισα σε αυτό, ήταν η προσαρμογή του αλγόριθμου *k*-Nearest-Neighbors στην δική μας περίπτωση, και συγκεκριμένα στην διαμόρφωση της εισόδου που έπρεπε να πάρει χρησιμοποιώντας τα δεδομένα που είχαμε. Τέλος, λόγω της κυβερνοεπίθεσης που έγινε κατά του Πανεπιστημίου Κύπρου, δεν κατάφερα να ανεβάσω το σύστημα στον web server για αρκετό καιρό, με αποτέλεσμα να μην έγιναν οι δοκιμές και το testing που ήθελα από διάφορους χρήστες.

Το αποτέλεσμα της εργασίας μου είναι ένα σύστημα συστάσεων που έχει όλες τις λειτουργίες όπως αναφέρθηκε πιο πάνω. Δηλαδή, με την χρήση των κατάλληλων δεδομένων και την εκτέλεση του αλγορίθμου, δίνονται συστάσεις με βάση αυτά. Στο σύστημα αυτό μπορούν να προστεθούν νέες λειτουργίες που μπορούν να το βελτιώσουν.

6.2 Εισηγήσεις για Μελλοντικές Εργασίες

Το σύστημα το οποίο αναπτύχθηκε μπορεί να τύχει βελτίωσης με την προσθήκη νέων λειτουργιών ή και την εφαρμογή κάποιων αναπροσαρμογών. Οι βελτιώσεις που μπορούν να γίνουν μπορεί να αφορούν την αποτελεσματικότητα του συστήματος, τον τρόπο λειτουργίας του, ή και άλλα θέματα. Ακολουθούν κάποιες προτάσεις για βελτίωση του συστήματος.

Χρήση User Authentication

Αυτή τη στιγμή, το σύστημα είναι προσβάσιμο από οποιονδήποτε χρήστη. Στο μέλλον όμως, θα μπορούσε να προστεθεί στο σύστημα μια μορφή user authentication, έτσι ώστε να έχουν πρόσβαση στο σύστημα μόνο χρήστες οι οποίοι θα επιτρέπεται, αν το σύστημα γίνει διαθέσιμο μόνο σε κάποιες επίλεκτες ομάδες (π.χ. μόνο ακαδημαϊκοί και φοιτητές). Οι χρήστες του συστήματος θα πρέπει να δημιουργούν προφίλ δίνοντας ένα username και ένα κωδικό, αλλά και την ιδιότητα τους, έτσι ώστε να τους δίνεται πρόσβαση στο σύστημα. Για να μπορεί να προστεθεί αυτή η λειτουργία, θα πρέπει να γίνει χρήση βάσης δεδομένων, που

δεν γίνεται τώρα, όπου θα αποθηκεύονται τα στοιχεία του κάθε χρήστη και θα γίνεται το authentication.

Χρήση ολόκληρων άρθρων ή της αρχικής περίληψης τους στον αλγόριθμο για λέξεις-κλειδιά αντί μόνο τίτλους

Όπως αναφέρθηκε και σε προηγούμενα κεφάλαια, στο σύστημα υλοποιείται ένας αλγόριθμος ο οποίος λαμβάνει υπόψη του λέξεις κλειδιά, και βρίσκει σε πόσους τίτλους καταχωρήσεων υπάρχουν. Υπάρχει η δυνατότητα ο αλγόριθμος αυτός να βελτιωθεί, αν εντοπίζει και τον αριθμό των φορών που εμφανίζονται οι λέξεις-κλειδιά μέσα σε ολόκληρα τα άρθρα που μας ενδιαφέρουν, ή στην αρχική τους περίληψη, με την χρήση του αλγόριθμου TF-IDF που ασχολείται με τη συχνότητα που εμφανίζονται λέξεις σε κείμενα. Αυτό δεν έγινε στο σύστημα που αναπτύχθηκε λόγω του ότι δεν υπήρχε πρόσβαση σε ολοκληρωμένα άρθρα, αλλά μόνο στους τίτλους τους, όμως αν βρεθεί τρόπος να υπάρχει πρόσβαση, τότε ένα τέτοιο στατιστικό όπως ο αριθμός ή το percentage των εμφανίσεων των λέξεων-κλειδιών μέσα στο άρθρο θα μπορούσε να βοηθήσει στην βελτιστοποίηση των αποτελεσμάτων. Ένα πιθανό αρνητικό που θα μπορούσε να προκληθεί με την εισαγωγή μιας τέτοιας λειτουργίας στον αλγόριθμο είναι η μείωση της απόδοσης του αλγόριθμου, που αν όμως δίνει καλύτερα αποτελέσματα και ο χρόνος εκτέλεσης δεν γίνει υπερβολικά πιο μεγάλος, τότε αξίζει τον κόπο.

Ενσωμάτωση καινούριων πηγών δεδομένων

Στο σύστημα που αναπτύχθηκε, τα δεδομένα που χρησιμοποιούνται από τον αλγόριθμο προέρχονται από τις εισόδους που παρέχει ο χρήστης και από την βάση δεδομένων του DBLP. Κάποιες πιθανές προσθήκες περισσότερων και διαφορετικών πηγών δεδομένων θα βοηθούσαν στην βελτίωση του συστήματος. Έχοντας ένα μεγαλύτερο όγκο δεδομένων που μπορούν να χρησιμοποιηθούν, ο αλγόριθμος μπορεί να γίνει καλύτερος δίνοντας καλύτερα και πιο σωστά αποτελέσματα, αφού θα έχει περισσότερα δεδομένα για να συγκρίνει με τις εισόδους.

Μηχανισμός Ανατροφοδότησης Χρηστών

Στο σύστημα θα μπορούσε στο μέλλον να προστεθεί ένας μηχανισμός για ανατροφοδότηση από τους χρήστες, με τον οποίο θα μπορούν οι χρήστες να δώσουν μια βαθμολογία αλλά και την γνώμη τους για τις συστάσεις που τους έγιναν, και γενικότερα για την εμπειρία που

έχουν όταν χρησιμοποιούν το σύστημα. Αυτά τα δεδομένα θα μπορούσαν να χρησιμοποιηθούν έτσι ώστε να βελτιωθεί ο αλγόριθμος και το σύστημα, αλλά και πιθανόν να βοηθήσουν στην καλύτερη εξατομίκευση των αποτελεσμάτων για κάθε χρήστη ανάλογα με τις εισόδους που δίνουν.

Χρήση Περιορισμών

Το σύστημα που αναπτύχθηκε θα μπορούσε να βελτιωθεί στο μέλλον με την εισαγωγή κάποιων περιορισμών στον αλγόριθμο. Για παράδειγμα, θα μπορούσαν να τεθούν κάποια χρονικά όρια μέσα στα οποία να πρέπει να είναι γραμμένα τα άρθρα που θα χρησιμοποιηθούν, και άρθρα που είναι γραμμένα εκτός αυτών των ορίων να μην λαμβάνονται υπόψη. Ακόμα, θα μπορούσαν να τεθούν περιορισμοί στον αριθμό άρθρων που θα πρέπει να έχει δημοσιεύσει κάποιος συγγραφέας για να μπορεί να προταθεί ως κριτής.

Δοκιμή Διαφορετικών Αλγόριθμων-Τεχνικών

Θα μπορούσαν στο μέλλον να δοκιμαστούν διαφορετικοί αλγόριθμοι μηχανικής μάθησης για εύρεση των ‘κοντινότερων’ συγγραφέων, όπως επίσης και διαφορετικές τεχνικές επεξεργασίας των δεδομένων που μπορεί να αποδειχθούν πιο αποδοτικοί και αποτελεσματικόι.

Βιβλιογραφία

[1] DBLP, https://dblp.org/

[2] Amatriain X., Jaimes A., Oliver N., Pujol J. M. "Data Mining Methods for Recommender Systems." *Recommender systems handbook*, Boston, MA: Springer US, pp. 39-71, 2010.

[3] HTML, https://developer.mozilla.org/en-US/docs/Web/HTML

[4] JavaScript, https://developer.mozilla.org/en-US/docs/Web/JavaScript

[5] Cascading Style Sheets, https://www.w3.org/Style/CSS/

[6] JSON (JavaScript Object Notation), https://www.json.org/json-en.html

[7] PHP, https://www.php.net/manual/en/intro-whatis.php

[8] Python, https://wiki.python.org/moin/BeginnersGuide/Overview

[9] JupyterLab, https://jupyterlab.readthedocs.io/en/stable/getting_started/overview.html

[10] Visual Studio Code, https://code.visualstudio.com/docs

[11] Khusro S., Ali Z., Ullah I., "Recommender Systems: Issues, Challenges, and Research Opportunities". *Information science and applications (ICISA)*, Springer Singapore, pp. 1179-1189, 2016.

[12] Decoding Amazon's Recommendation System, https://www.argoid.ai/blog/decoding-amazons-recommendation-system

[13] How Netflix's Recommendations System Works, https://help.netflix.com/en/node/100639#:~:text=We%20estimate%20the%20likelihood%20that,preferences%20on%20our%20service%2C%20and

[14] Inside Spotify's Recommender System: A Complete Guide to Spotify Recommendation Algorithms, https://www.music-tomorrow.com/blog/how-spotify-recommendation-system-works-a-complete-guide-2022

[15] Van Meteren R., Van Someren M., "Using Content-Based Filtering for Recommendation.", *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop*, vol. 30, pp. 47-56, 2000.

[16] Bafna P., Pramod D., Vaidya A., "Document clustering: TF-IDF approach." *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, IEEE, pp. 61-66, 2016.

[17] Herlocker J. L., Konstan J. A., Riedl J., "Explaining Collaborative Filtering recommendations.", *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pp. 241-250, 2000.

[18] Adomavicius G., Tuzhilin A., "Context-aware recommender systems.", *Recommender systems handbook*, Boston, MA: Springer US, pp.. 217-253, 2010.

[19] What Is Agile Software Development? Life Cycle, Methodology, and Examples, https://www.spiceworks.com/tech/devops/articles/what-is-agile-software-development/

[20] Demystifying the Parquet File Format, https://towardsdatascience.com/demystifying-the-parquet-file-format-13adb0206705