

Ιόνιο Πανεπιστήμιο

Τμήμα Πληροφορικής



Πτυχιακή Εργασία

**Εφαρμογή web για τη δημιουργία βιβλιοθήκης εγγράφων με ανάλυση
συναισθήματος και αναγνώριση οντοτήτων**

Φάμπιαν Χαίγκερ

Επιβλέπων: Μιχάλης Στεφανιδάκης

8 Σεπτεμβρίου 2021

**Τίτλος: Εφαρμογή web για τη δημιουργία βιβλιοθήκης εγγράφων με
ανάλυση συναισθήματος και αναγνώριση οντοτήτων**

Συγγραφέας: Φάμπιαν Χαίγκερ

Τριμελής επιτροπή:

Μιχάλης Στεφανιδάκης, Καθηγητής

Ελένη Χριστοπούλου, Εργαστηριακό διδακτικό προσωπικό

Θεόδωρος Ανδρόνικος, Αναπληρωτής καθηγητής

Περίληψη

Ο τίτλος της αναφοράς είναι Εφαρμογή web για τη δημιουργία βιβλιοθήκης εγγράφων με ανάλυση συναισθήματος και αναγνώριση οντοτήτων. Περιγράφεται η ανάπτυξη διαδικτυακής εφαρμογής με δυνατότητες ανάλυσης, επεξεργασίας και αποθήκευσης άρθρων. Στο κεφάλαιο 1 περιγράφονται οι λειτουργίες της εφαρμογής και ο σκοπός δημιουργίας της. Στο κεφάλαιο 2 γίνεται συνοπτική περιγραφή της θεωρίας διαφόρων αλγορίθμων και τεχνικών που χρησιμοποιήθηκαν στην εφαρμογή. Στο 3ο κεφάλαιο κάθε ενότητα αναφέρεται συνοπτικά σε μια βιβλιοθήκη κώδικα που έχει χρησιμοποιηθεί στην εφαρμογή. Αποτελείται συνολικά από έντεκα ενότητες. Το κεφάλαιο 4 αποτελείται από τέσσερις ενότητες και σκιαγραφεί τη διαδικασία ανάπτυξης των μοντέλων μηχανικής μάθησης που χρησιμοποιήθηκαν στην εφαρμογή. Οι πρώτες τρεις ενότητες αφορούν την δημιουργία μοντέλου ανάλυση συναισθήματος σε σώμα κειμένου, ενώ η τέταρτη ανάπτυξη μοντέλου εύρεσης παρόμοιων κειμένων. Το κεφάλαιο πέντε αποτελείται από έντεκα ενότητες. Σε αυτό αναλύεται λεπτομερώς η αρχιτεκτονική και τα δομικά στοιχεία της εφαρμογής. Αρχικά αναλύεται η βάση δεδομένων και η γενική αρχιτεκτονική της εφαρμογής. Στη συνέχεια περιγράφονται οι σελίδες της εφαρμογής και τα πιο σημαντικά μέρη του κώδικα. Η τελευταία ενότητα περιέχει οδηγίες εγκατάστασης της εφαρμογής. Στο τελευταίο κεφάλαιο αναφέρονται τα συμπεράσματα και πιθανές μελλοντικές βελτιώσεις της εφαρμογής.

The title of this essay is web application for sentiment analysis, entity recognition and archiving of news articles. The first chapter explains the reasons for the creation of this web application its use. In the second chapter there is a short introduction to the techniques and algorithms used to achieve the goals of the web application. The third chapter traces the various bibliography used to implement all the functions of the web application. The fourth chapter contains all the steps that were necessary in order to create the models used for sentiment analysis and topic modeling tasks. The fifth chapter consists of the detailed description of every important part of the web application. Its initial part describes the architecture of the database, the middle section outlines the front end of the application and, finally, in the last segment the most important parts of the code base are illustrated. There are also instructions for readers who would like to install the application in their own system. The last chapter includes various possible improvements of the application and some deductions about the use of machine learning in modern society.

Περίληψη	2
Κεφάλαιο 1 Εισαγωγή	6
Κεφάλαιο 2 Θεωρητικό υπόβαθρο της εφαρμογής	7
2. 1 Sentiment Analysis	7
2. 2 Topic Modelling	7
2. 3 Αναγνώριση οντοτήτων	8
2. 4 Latent dirichlet allocation	8
2. 5 Lemmatization	9
2. 6 Stemming	9
2. 7 Count-Vectorization	9
2. 8 Term frequency–inverse document frequency vectorization	9
2. 9 Recurrent Neural Network	10
2. 10 Random Forests	10
2. 11 Levenshtein distance	10
Κεφάλαιο 3 Βιβλιοθήκες κώδικα	11
3.1 Django	11
3.2 Tensorflow	11
3.3 Numpy	12

	5
3.4 Scrapy	12
3.5 BeautifulSoup	12
3.6 NLTK	13
3.7 Spacy	13
3.8 Gensim	13
3.9 fuzzywuzzy	13
3.10 Wikipedia	13
3.11 Pickle	14
Κεφάλαιο 4 Ανάπτυξη του μοντέλου για Sentiment Analysis	15
4. 1 ΣΕΤ Δεδομένων	15
4. 2 Random Forests	18
4. 3 Recurrent neural network	21
Combination of reviews	22
4. 4 Ανάπτυξη του LDA	27
Κεφάλαιο 5 Η εφαρμογή	35
5. 1 Βάση δεδομένων	35
5. 2 Object Relational Mapper	40
5. 3 Αρχική Σελίδα	42
5. 4 Βιβλιοθήκες	44
5. 5 Άρθρο	46
5. 6 Χρήστες	47
5. 7 Σελίδα υποβολής άρθρων	51
5. 8 Οντότητα	54
5. 9 Δεδομένα	54
5. 10 Πληροφορίες Εφαρμογής	58
5. 11 Επεξήγηση Κώδικα	59
5.12 Εγκατάσταση Εφαρμογής	67
Κεφάλαιο 6 Συμπεράσματα και Μελλοντικές επεκτάσεις	68
Bibliography	69

Κεφάλαιο 1 Εισαγωγή

Η εφαρμογή που περιγράφεται παρακάτω, δημιουργήθηκε με σκοπό τη βοήθεια του αναγνώστη, στον απώτερο σκοπό που είναι η σωστή ενημέρωση. Δίνοντας περισσότερες πληροφορίες όχι μόνο για τις ειδήσεις και τα πρόσωπα που αφορούν αλλά και τις πηγές των ειδήσεων, διευρύνεται η αντίληψη του αναγνώστη για τη φύση των ειδήσεων που επιλέγει να καταναλώσει. Χρησιμοποιώντας πρόσφατα τεχνολογικά μέσα έγινε προσπάθεια συλλογής σημαντικών πληροφοριών άρθρων και μεταφορά τους στο χρήστη της εφαρμογής. Ακόμη δίνεται η δυνατότητα στους χρήστες αποθήκευσης των άρθρων. Όπως είναι γνωστό, είναι συχνά εύκολο άρθρα παλαιότερων περιόδων να γίνουν δυσεύρετα με την πάροδο του χρόνου λόγω της φύσεως των περισσότερο χρησιμοποιούμενων μηχανών αναζήτησης που εμφανίζουν κυρίως τα πιο πρόσφατα αποτελέσματα .

Η φύση των πληροφοριών περιλαμβάνει τις πηγές των άρθρων και τις οντότητες που αναφέρονται στα άρθρα. Η εφαρμογή μπορεί επίσης, να χρησιμοποιηθεί για την ανάγνωση των ήδη καταχωρημένων άρθρων. Στη συνέχεια ακολουθεί μια περιγραφή τεχνικών και αλγορίθμων που χρησιμοποιήθηκαν στην εφαρμογή.

Κεφάλαιο 2 Θεωρητικό υπόβαθρο της εφαρμογής

2. 1 Sentiment Analysis

Sentiment Analysis, ονομάζεται η προσπάθεια κατηγοριοποίησης ενός κειμένου φυσικής γλώσσας, με βάση τον τόνο ή το συναίσθημα του κειμένου. Συνήθως τα κείμενα αυτά κατηγοριοποιούνται σε θετικά, ουδέτερα ή αρνητικά, αν και κατά διαστήματα χρησιμοποιούνται περαιτέρω υποκατηγορίες.

Το Sentiment Analysis ανήκει στον χώρο του NLP (Natural Language Processing) και οι τεχνικές που χρησιμοποιούνται για την επίτευξή του διαφέρουν κατά πολύ, καθώς αποτελεί τομέα πολύ ενεργής έρευνας στον ακαδημαϊκό κόσμο. Κάποιες από τις πιο γνωστές τεχνικές αποτελούνται από μοντέλα μηχανικής μάθησης με αλλά και χωρίς επίβλεψη. Η έρευνα στο τομέα αυτό ξεκίνησε της αρχές του 2000, με τους Bo Pang, Lillian Lee και Peter Turney. (Bo Pang-Lillian Lee-Peter Turney, 2002)

2. 2 Topic Modelling

Topic Modelling, είναι η τεχνική διαχωρισμού ενός σώματος κειμένων σε μικρότερες θεματικές ομάδες, με κριτήριο τη συχνότητα εμφάνισης των λέξεων. Τα μοντέλα που ανήκουν σε αυτόν τον τομέα, λειτουργούν υπό την προϋπόθεση ότι κάποια κείμενα έχουν κοινές λέξεις οι οποίες δεν εμφανίζονται με την ίδια συχνότητα σε άλλα κείμενα. Κείμενα με κοινές λέξεις ανήκουν στην ίδια θεματική ενότητα.

Η παραπάνω υπόθεση εφαρμόζεται με μαθηματικούς όρους, με τη χρήση πιθανοτήτων και τη δημιουργία μέτρων απόστασης. Γενικότερα, οι αλγόριθμοι αυτοί ανήκουν στην ευρύτερη κατηγορία των αλγορίθμων ομαδοποίησης ή clustering algorithms, όπου τα δεδομένα διαχωρίζονται σε μη προκαθορισμένες ομάδες, σε αντίθεση με μοντέλα όπου τα δεδομένα ταξινομούνται σε συγκεκριμένες κατηγορίες.

2.3 Αναγνώριση οντοτήτων

Η αναγνώριση οντοτήτων εντός ενός κειμένου και η κατηγοριοποίησή του είναι τεχνική που προήλθε από τον τομέα της ανάκτησης πληροφορίας και πλέον πολλές τεχνικές αντιμετώπισης αυτού του προβλήματος προέρχονται από το χώρο του NLP.

2.4 Latent dirichlet allocation

Latent dirichlet allocation, ονομάζεται ο αλγόριθμος που χρησιμοποιήθηκε για την εύρεση των θεμάτων που εμφανίζονται στα άρθρα, που έχουν αποθηκευτεί στην εφαρμογή. Ο αλγόριθμος αυτός, δημιουργήθηκε από τους J. K. Pritchard, M. Stephens και P. Donnelly, το 2000 (Jonathan K. Pritchard-Matthew Stephens-Peter Donnelly, 2000) με σκοπό την ανάλυση γενετικών δεδομένων, αλλά στη συνέχεια χρησιμοποιήθηκε στο χώρο της μηχανικής μάθησης και πιο συγκεκριμένα, του NLP.

Ο αλγόριθμος αναπαριστά ένα σώμα κειμένων, ως ένα πίνακα λέξεων και κειμένων και στη συνέχεια δημιουργεί δύο νέους πίνακες: έναν για την αντιστοίχιση θεμάτων με κείμενα και έναν για την αντιστοίχιση θεμάτων με λέξεις. Οι πίνακες αυτοί δημιουργούνται σύμφωνα με στατιστικά στοιχεία, για τις λέξεις που αντλούνται από το σώμα κειμένων. Η αξιολόγηση του μοντέλου δεν μπορεί να γίνει με κάποιου είδους μέτρο, όπως η ακρίβεια, διότι το σώμα κειμένων δεν αποτελεί "σετ" δεδομένων, με παραδείγματα που ανήκουν σε κάποια καθορισμένη κατηγορία. Υπάρχουν κάποια άλλα μέτρα που έχουν δημιουργηθεί για τον λόγο αυτό, με κύριο παράδειγμα το U-mass, C_v, καθώς και αλλά πολλά. Με το μοντέλο αυτό, εκτός από την εύρεση των θεμάτων που υπάρχουν στο κείμενο, είναι δυνατό με την χρήση ορισμένων μέτρων ομοιότητας, να γίνει εύρεση κειμένων με παρεμφερή θεματολογία, είτε αυτό πρόκειται για ένα νέο κείμενο, είτε για ένα κείμενο του αρχικού σώματος.

2. 5 Lemmatization

Lemmatization, είναι η διαδικασία με την οποία κάθε λέξη του κειμένου αντικαθίστανται με το λήμμα της (ο βασικός τύπος κάθε λέξης σ' ένα λεξικό). Η διαδικασία αυτή, χρησιμοποιείται συχνά στον χώρο του NLP διότι διευκολύνει την εκπαίδευση των μοντέλων, καθώς μειώνεται ο όγκος των δεδομένων, διατηρώντας την σημαντική πληροφορία.

2. 6 Stemming

Το Stemming μοιάζει με το Lemmatization, με την διαφορά ότι αντί οι λέξεις να αντικαθίστανται με το λήμμα τους, γίνεται μια πιο απλή διαδικασία, όπου αφαιρούνται οι καταλήξεις των λέξεων. Αυτό βέβαια σημαίνει ότι σε κάποιες περιπτώσεις αφαιρείται υπερβολική πληροφορία και δυο λέξεις που δεν σχετίζονται μεταξύ τους καταλήγουν να έχουν ίδια μορφή. Παρόλα αυτά, το stemming χρησιμοποιείται κατά κύριο λόγο, διότι είναι ταχύτερο από το Lemmatization.

2. 7 Count-Vectorization

Το Count-Vectorization(CV), είναι μια διαδικασία, όπου η κάθε λέξη του σώματος κειμένων υπολογίζεται ως το πλήθος των εμφανίσεων της, εντός του σώματος. Κατά τη διαδικασία αυτή, αντικαθίσταται κάθε λέξη στο "σετ" δεδομένων με το CV της λέξης.

2. 8 Term frequency–inverse document frequency vectorization

Το tf-idf(tfidf-Vectorization) ως μέτρο, έχει πολλές εφαρμογές σε διαφορετικούς τομείς. Ορίζεται ως $tfidf(t,d,D)=tf(t,d)*idf(t,D)$, όπου tf είναι η συχνότητα μια λέξης στο κείμενο. Idf ορίζεται ως το κλάσμα του αριθμού κειμένων προς τον αριθμό κειμένων που περιέχουν τον όρο, σε λογαριθμική κλίμακα. Δημιουργήθηκε από τους (KarenSpärckJones, 1972) και (GeraldStalton, 1972).

Στον χώρο του NLP, χρησιμοποιείται στην επεξεργασία κειμένου. Οι λέξεις του σώματος κειμένου, αντικαθίστανται με την τιμή tf-idf ,για την συγκεκριμένη λέξη. Έτσι, “διευκολύνεται” το μοντέλο στην εύρεση μοτίβων, καθώς το κείμενο στην αρχική του μορφή δεν είναι κατανοητό.

2. 9 Recurrent Neural Network

Τα RNN (Recurrent Neural Network), αποτελούν κατηγορία τεχνητών νευρωνικών δικτύων. Βασικό στοιχείο ενός δικτύου RNN, είναι το επίπεδο LSTM (Long short-term memory). Δημιουργήθηκαν από τον John Joseph Hopfield.

Το LSTM είναι σχεδιασμένο έτσι ώστε να διατηρεί ένα είδος μνήμης και να είναι έτσι δυνατή η επεξεργασία δεδομένων διαδοχικής φύσης, όπως ο γραπτός ή προφορικός λόγος.

2. 10 Random Forests

Το Random Forests είναι αλγόριθμος κατηγοριοποίησης. Δημιουργήθηκε από (Ho, 1995) το 1995 και χρησιμοποιείται ακόμα και σήμερα. Βασίζεται σε ένα πλήθος ανεξάρτητων δέντρων αποφάσεων, που στο σύνολό τους επιτυγχάνει την κατηγοριοποίηση των δεδομένων.

2. 11 Levenshtein distance

Το Levenshtein distance είναι μέτρο σύγκρισης αλληλουχιών. Η απόσταση μεταξύ δύο λέξεων, ορίζεται ως ο ελάχιστος αριθμός αλλαγών γραμμάτων, για την αλλαγή της μιας λέξης σε μια άλλη.

Κεφάλαιο 3 Βιβλιοθήκες κώδικα

3.1 Django

Το Django, είναι βιβλιοθήκη που υποστηρίζει τη δημιουργία ιστοσελίδων δυναμικού περιεχομένου με χρήση της γλώσσας python. Το Django αναλαμβάνει όλη την διαδικασία δημιουργίας δυναμικών ιστοσελίδων, από την τοποθέτηση του περιεχομένου εντός της ιστοσελίδας, μέσω της χρήσης μιας γλώσσας σήμανσης, έως τη σύνδεση της σελίδας, με μια βάση δεδομένων.

Αυτή η ολιστική προσέγγιση έρχεται σε αντίθεση με τις πιο διαδεδομένες πρακτικές στον χώρο της ανάπτυξης εφαρμογών, που περιλαμβάνουν τον συνδυασμό διαφορετικών προγραμμάτων εξειδικευμένων σε συγκεκριμένους σκοπούς.

Η εγκατάσταση του μπορεί να γίνει μέσω της εντολής `python -m pip install Django` σε ένα virtual environment της επιλογής μας.

3.2 Tensorflow

Το Tensorflow είναι μια βιβλιοθήκη που κάνει δυνατή την ανάπτυξη μοντέλων μηχανικής μάθησης. Μπορεί να χρησιμοποιηθεί από αρχάριους αλλά και έμπειρους επαγγελματίες, καθώς παρέχει την δυνατότητα να αναπτυχθούν μοντέλα με απόλυτο έλεγχο κάθε παραμέτρου, αλλά και διευκολύνσεις που δίνουν την δυνατότητα ανάπτυξης αποτελεσματικών μοντέλων σε λίγες γραμμές κώδικα. Πιο συγκεκριμένα, το tensorflow παρέχει API για την ανάπτυξη της πλειοψηφίας των μοντέλων μηχανικής μάθησης που έχουν δημιουργηθεί από την ακαδημαϊκή κοινότητα, διευκολύνσεις για την επεξεργασία δεδομένων, την απεικόνιση των αποτελεσμάτων, αλλά και την αποθήκευση και χρήση των μοντέλων σε πραγματικές εφαρμογές.

Η εγκατάσταση του tensorflow διαφέρει από εκδοχή σε εκδοχή. Υπάρχει tensorflow cpu η οποία χρησιμοποιεί τον επεξεργαστή του υπολογιστή και είναι πιο εύκολο να χρησιμοποιηθεί στη πλειοψηφία των σύγχρονων υπολογιστών. Η εκδοχή του tensorflow που λειτουργεί στην κάρτα γραφικών έχει περισσότερες απαιτήσεις και εξαρτάται από τον υπολογιστή του χρήστη.

3.3 Numpy

Το numpy, είναι μια βιβλιοθήκη βελτιστοποιημένη για την ανάπτυξη υπολογισμών, στον χώρο των πράξεων πινάκων μέσω της γεωμετρικής άλγεβρας, την οποία χρησιμοποιούν πολλά μοντέλα μηχανική μάθησης, αλλά και άλλες ειδικότητες, όπως λ. χ. η βιοπληροφορική που βασίζεται σε τέτοιους είδους υπολογισμούς.

Το numpy είναι διαθέσιμο για εγκατάσταση με πολλούς τρόπους. Μπορεί να εγκατασταθεί μέσω του Conda αλλά και του pip.

3.4 Scrapy

Το scrapy, είναι βιβλιοθήκη για τη γλώσσα Python, που παρέχει τη δυνατότητα αυτοματοποίησης της συλλογής δεδομένων από ιστοσελίδες και άλλες ηλεκτρονικές πηγές.

Για την αποτελεσματική συλλογή πληροφοριών, είναι ωφέλιμο η διαδικασία να είναι αυτοματοποιημένη και η ίδια αντιμετώπιση να λειτουργεί σε διαφορετικές πηγές. Αυτό ακριβώς επιτυγχάνει το scrapy, προσφέροντας κλάσεις που επιτρέπουν την προσπέλαση εκατοντάδων ιστοσελίδων σε μερικά λεπτά και την αποθήκευση της πληροφορίας που πληροί τα κριτήρια που έχει προκαθορίσει ο χρήστης νωρίτερα.

Η εγκατάσταση του scrapy διαφέρει αναλόγως με το λογισμικό του χρήστη. Για χρήστες Windows συνιστάται η εγκατάσταση μέσω του Conda ενώ για χρήστες Linux είναι διαθέσιμο και μέσω pip.

3.5 BeautifulSoup

Το beautifulsoup, είναι μια ευρέως γνωστή βιβλιοθήκη για τη συλλογή δεδομένων, από αρχεία html και xml. Η εγκατάσταση του γίνεται μέσω της εντολής `pip install beautifulsoup4`.

3.6 NLTK

Το nltk διευκολύνει την πρόσβαση σε διάφορους πόρους, όπως δεδομένα και αλγόριθμους επεξεργασίας, στον χώρο του Natural language processing και χρησιμοποιείται κυρίως στην εκμάθηση των τεχνικών αυτών. Το nltk μπορεί να εγκατασταθεί σε όλα τα λειτουργικά συστήματα αν και η διαδικασία ενδέχεται να διαφέρει.

3.7 Spacy

Το spacy, είναι μια βιβλιοθήκη με μια συλλογή μοντέλων και κλάσεων για την ανάπτυξη μοντέλων μηχανικής μάθησης, στο χώρο της φυσικής γλώσσας, γνωστό και ως NLP (NaturalLanguageProcessing). Το spacy παρέχει πρόσβαση σε όλους σε πολύ ισχυρά μοντέλα, ήδη εκπαιδευμένα σε τεράστιο όγκο δεδομένων, τα οποία διαφορετικά δεν θα ήταν δυνατό να χρησιμοποιηθούν από την πλειοψηφία του κόσμου. Ακόμη, περιέχει την δυνατότητα περαιτέρω εκπαίδευσης των υπάρχοντων μοντέλων για την εξειδίκευσή τους στον σκοπό του χρήστη και τα μοντέλα αυτά είναι διαθέσιμα για 19 (δεκαεννιά) γλώσσες.

Η εγκατάσταση του spacy μπορεί να γίνει μέσω pip και Conda και, ανάλογα με τη χρήση, μπορούν να εγκατασταθούν διαφορετικά πακέτα.

3.8 Gensim

Το gensim, είναι μια βιβλιοθήκη στο χώρο του NLP, εξειδικευμένη στην ανάπτυξη μοντέλων εξόρυξης δεδομένων από κείμενα φυσικής γλώσσας. Η εγκατάσταση του μπορεί να γίνει μέσω της εντολής pip install gensim.

3.9 fuzzywuzzy

Το fuzzywuzzy παρέχει κάποιους αλγόριθμους για την εύρεση ομοιότητας λέξεων σε γλώσσα Python. Η εγκατάσταση του μπορεί να γίνει μέσω της εντολής pip install fuzzywuzzy.

3.10 Wikipedia

Η βιβλιοθήκη wikipedia, παρέχει ένα API για την αλληλεπίδραση με τη γνωστή συνονόματη πηγή δεδομένων. Η εγκατάσταση του μπορεί να γίνει μέσω της εντολής pip install wikipedia.

3.11 Pickle

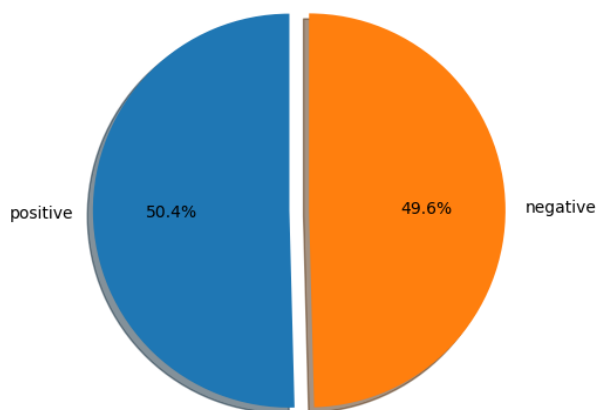
Το pickle, μετατρέπει αντικείμενα της γλώσσας python, από και σε σειριακά αρχεία. Η εγκατάσταση του μπορεί να γίνει μέσω της εντολής `pip install pickle`.

Κεφάλαιο 4 Ανάπτυξη του μοντέλου για Sentiment Analysis

4.1 Σετ Δεδομένων

Κατά την ανάπτυξη του μοντέλου για Sentiment Analysis χρησιμοποιήθηκαν τρία διαφορετικά "σετ" επισυναπτόμενων δεδομένων για την σύγκριση των αποτελεσμάτων.

Αρχικά, χρησιμοποιήθηκε μια συλλογή κριτικών από το Amazon, imdb και το Yelp. Τα δεδομένα αποτελούνται από περίπου 3000 (τρεις χιλιάδες) κριτικές, ταξινομημένες σε θετικές και αρνητικές με παρόμοια κατανομή όπως φαίνεται και στο ακόλουθο σχήμα.



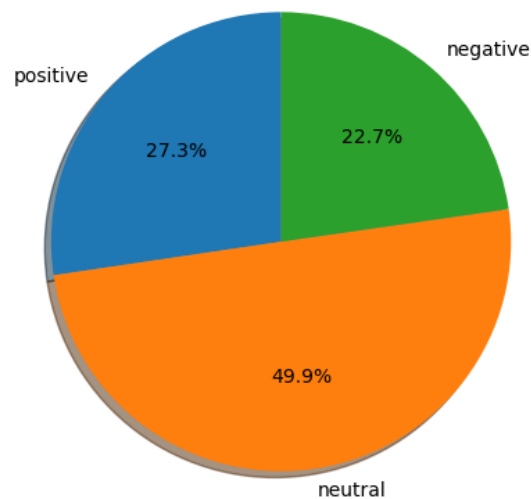
Σχήμα 1 Combination of reviews dataset

Ακόμη, χρησιμοποιήθηκε ένα "σετ" δεδομένων προερχόμενο από το πανεπιστήμιο του Stanford, γνωστό ως (StanfordSentimentTreebank). Τα δεδομένα αποτελούνται από 24.000 (είκοσι τέσσερις χιλιάδες) κριτικές ταινιών, συλλεγμένες από τη γνωστή ιστοσελίδα Rotten Tomatoes.

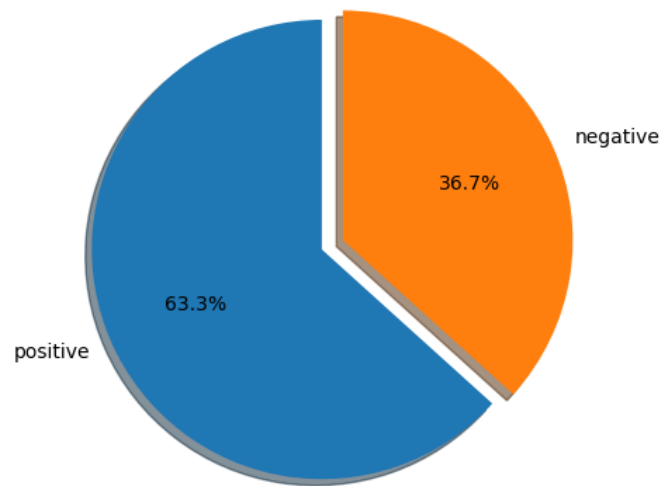
Τα δεδομένα είναι ταξινομημένα με μια τιμή από το 0 (μηδέν) ως το 1 (ένα), όπου 0-0.2 σημαίνει πολύ αρνητικό, 0.2-0.4 σημαίνει αρνητικό και 0.4-0.6 σημαίνει

ουδέτερο. Αντίστοιχα 0.6-0.8 θετικό και 0.8-1.0 πολύ θετικό. Για τις δοκιμές στο συγκεκριμένο "σετ" δεδομένων, έγινε μια απλοποίηση όπου οι κριτικές με τιμή 0-0.4 θεωρήθηκαν αρνητικές, 0.4-0.6 ουδέτερες και 0.6-1 θετικές. Επίσης, σε κάποιες δοκιμές χρησιμοποιήθηκαν μόνο οι αρνητικές και θετικές κριτικές.

Παρακάτω βρίσκονται οι αντίστοιχες κατανομές. Είναι φανερό ότι οι δυο κλάσεις έχουν αρκετή διαφορά στο πλήθος τους, κάτι που ενδέχεται να προκαλέσει μια προτίμηση προς τα θετικά αποτελέσματα, στα παραγόμενα μοντέλα.

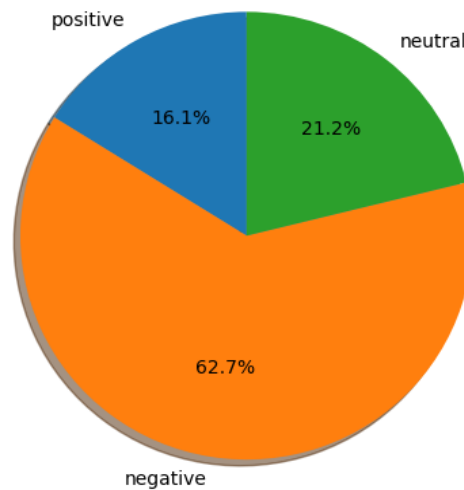


Σχήμα1 Stanford Sentiment Treebank Dataset



Σχήμα2 Stanford Sentiment Treebank with 2 classes

Το τρίτο “σετ” δεδομένων, συλλέχθηκε από το κοινωνικό δίκτυο Twitter και αποτελείται κατά κύριο λόγο από περίπου από 15. 000 (δεκαπέντε χιλιάδες) tweets προς αεροπορικές εταιρείες, τα οποία έχουν ταξινομηθεί σε αρνητικά, θετικά και ουδέτερα (Twitter US Airline Sentiment) (StanfordSentimentTreebank). Το “σετ” δεδομένων περιέχει και κάποιες επιπλέον πληροφορίες, που δεν χρησιμοποιήθηκαν. Το συγκεκριμένο “σετ” έχει αρκετά μεγάλη δυσαναλογία στο πλήθος των παραδειγμάτων μεταξύ των κλάσεων.



Σχήμα3 Twitter Airline Dataset

4. 2 Random Forests

Με τη χρήση των δεδομένων αυτών, έγινε η δοκιμή δυο ειδών διαφορετικών μοντέλων, όπως και τον συνδυασμό διαφορετικών μεθόδων επεξεργασίας. Παρακάτω, βρίσκονται αριθμημένες οι μέθοδοι και οι παράμετροι για τον αλγόριθμο Random Forests. (n_est αναφέρεται στον αριθμό των estimators και max_depth στο μέγιστο βάθος του δέντρου).

1. **CountVectorizer(CV)--Stemming--n_est=50, max_depth=20**
2. CV--Lemmatizing--n_est=50, max_depth=20
3. CV--Stemming--n_est=150, max_depth=None
4. CV--Lemmatizing--n_est=150, max_depth=None
5. **TfidfVectorizer(tfidf)--Stemming--n_est=50, max_depth=20**
6. tfidf--Lemmatizing--n_est=50, max_depth=20
7. tfidf--Stemming--n_est=150, max_depth=None
8. tfidf--Lemmatizing--n_est=150, max_depth=None

Αποτελέσματα

Combination of Reviews

Metrics	#1	#2	#3	#4	#5	#6	#7	#8
Precision	0.66	0.78	0.79	0.81	0.78	0.80	0.88	0.80
Recall	0.85	0.64	0.79	0.64	0.70	0.65	0.73	0.65
Accuracy	0.72	0.73	0.80	0.75	0.75	0.73	0.80	0.73
Preprocessing	CV	CV	CV	CV	tfidf	tfidf	tfidf	tfidf

Σχήμα4 Combination of Reviews RF results

Stanford Dataset

Metrics	#1	#2	#3	#4	#5	#6	#7	#8
Precision	0.61	0.6	0.89	0.89	0.74	0.75	0.81	0.8
Recall	0.98	0.98	0.94	0.93	0.88	0.88	0.88	0.88
Accuracy	0.65	0.64	0.9	0.9	0.77	0.78	0.82	0.82
Preprocessing	CV	CV	CV	CV	tfidf	tfidf	tfidf	tfidf

Σχήμα5 Stanford Sentiment Treebank RF results

Twitter Airline Dataset

Metrics	#1	#2	#3	#4	#5	#6	#7	#8
---------	----	----	----	----	----	----	----	----

Precision	1	1	0. 86	0. 86	0. 96	1	0. 84	0. 8
Recall	0. 006	0. 002	0. 62	0. 64	0. 17	0. 01	0. 63	0. 61
Accuracy	0. 79	0. 78	0. 89	0. 9	0. 83	0. 8	0. 9	0. 89
Preproce ssing	CV	CV	CV	CV	tfidf	tfidf	tfidf	tfidf

Σχήμα6 Twitter Airline Dataset RF results

Κάποια συμπεράσματα που μπορούν να εξαχθούν είναι αρχικά ότι δεν υπάρχει μεγάλη διαφορά στα αποτελέσματα στα οποία χρησιμοποιήθηκε Stemming από εκείνα όπου χρησιμοποιήθηκε Lemmatizing.

Πιο συγκεκριμένα, συγκρίνοντας την στήλη 1 με την στήλη 2 και αντιστοίχως την στήλη 3 με την στήλη 4, δεν υπάρχει σημαντική διαφορά. Αντίστροφα, συγκρίνοντας τις στήλες 1 και 3 και ακολούθως τις στήλες 2 και 4, γίνεται αντιληπτό πως υπάρχει εμφανής διαφορά στα μέτρα Precision και Accuracy. Αυτό ίσως δείχνει ότι η διαφορά στις παραμέτρους έχει μεγαλύτερο αντίκτυπο στην απόδοση του μοντέλου από τις μεθόδους προεπεξεργασίας, στην προκειμένη περίπτωση. Την υπόθεση αυτή ενισχύει το γεγονός ότι η προαναφερόμενη τάση παρατηρείται και στα τρία “σετ” δεδομένων που προέρχονται από πολύ διαφορετικές πηγές, κάτι το οποίο είναι εξαιρετικής σημασίας, στο χώρο του natural language processing.

Μια επιπλέον παρατήρηση είναι το γεγονός ότι, ενώ στα πρώτα δυο “σετ” δεδομένων οι τιμές των μέτρων αυξάνονται με την αύξηση του μεγέθους του μοντέλου (δηλαδή με την αύξηση του βάθους και του πλήθους των estimators), στο τελευταίο “σετ” δεδομένων συμβαίνει το αντίθετο. Πιθανολογείται ότι αυτό μπορεί να οφείλεται στο γεγονός ότι τα μικρότερα μοντέλα, επηρεάζονται περισσότερο από τη δυσαναλογία στο πλήθος των παραδειγμάτων που είχε αναφερθεί παραπάνω και η αυξημένη ακρίβεια, ενδεχομένως δεν οφείλεται σε επιτυχημένη πρόβλεψη, αλλά στο γεγονός ότι το μοντέλο ταξινομεί τη πλειοψηφία των παραδειγμάτων ως αρνητικά.

4. 3 Recurrent neural network

Παρακάτω εμφανίζονται οι δυο διαφορετικές αρχιτεκτονικές του RNN, που χρησιμοποιήθηκαν.

Στο πρώτο νευρωνικό δίκτυο χρησιμοποιήθηκαν, εκτός του αρχικού στρώματος 32 πυρήνων, ένα στρώμα LSTM και στη συνέχεια, δυο πυκνά στρώματα επίσης 32 πυρήνων. Ως activation function χρησιμοποιήθηκε η Softmax. Activation functions είναι συναρτήσεις που χρησιμοποιούνται στην μηχανική μάθηση ως μέθοδοι επεξεργασίας των δεδομένων που δέχονται οι κόμβοι των τεχνητών νευρωνικών δικτύων. Η συνάρτηση Softmax ορίζεται ως

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

Όπου z διάνυσμα εισόδου στη συνάρτηση. z_i είναι αριθμός του διανύσματος και K ο αριθμός κλάσεων στις οποίες πρέπει να κατηγοριοποιηθούν τα δεδομένα. Το δίκτυο είχε ως σύνολο 446, 081 παραμέτρους.

Στο δεύτερο νευρωνικό δίκτυο αυξήθηκε το μέγεθος σε 64 πυρήνες ανά στρώμα και η Softmax αντικαταστάθηκε με την Sigmoid, που έχει συχνότερα καλύτερα αποτελέσματα σε προβλήματα δύο κλάσεων. Η Sigmoid ή σιγμοειδής ορίζεται ως $S(t)=1/(1+e^{-t})$.

Ως μέθοδος εκπαίδευσης χρησιμοποιήθηκε η 5 fold Cross Validation. K-fold Cross Validation ονομάζεται η μέθοδος εκπαίδευσης όπου το μοντέλο μηχανικής μάθησης εκπαιδεύεται ξανά και ξανά, κάθε φορά σε διαφορετικό μέρος του συνόλου του σετ δεδομένων, αφήνοντας πάντα ένα μέρος των δεδομένων μόνο για επαλήθευση των αποτελεσμάτων.(M. Stone , 1974)

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 32)	436672
lstm_1 (LSTM)	(None, 32)	8320
dense_1 (Dense)	(None, 32)	1056
dense_2 (Dense)	(None, 1)	33
Total params: 446,081		
Trainable params: 446,081		
Non-trainable params: 0		

Σχήμα 7 1η Αρχιτεκτονική μοντέλου RNN

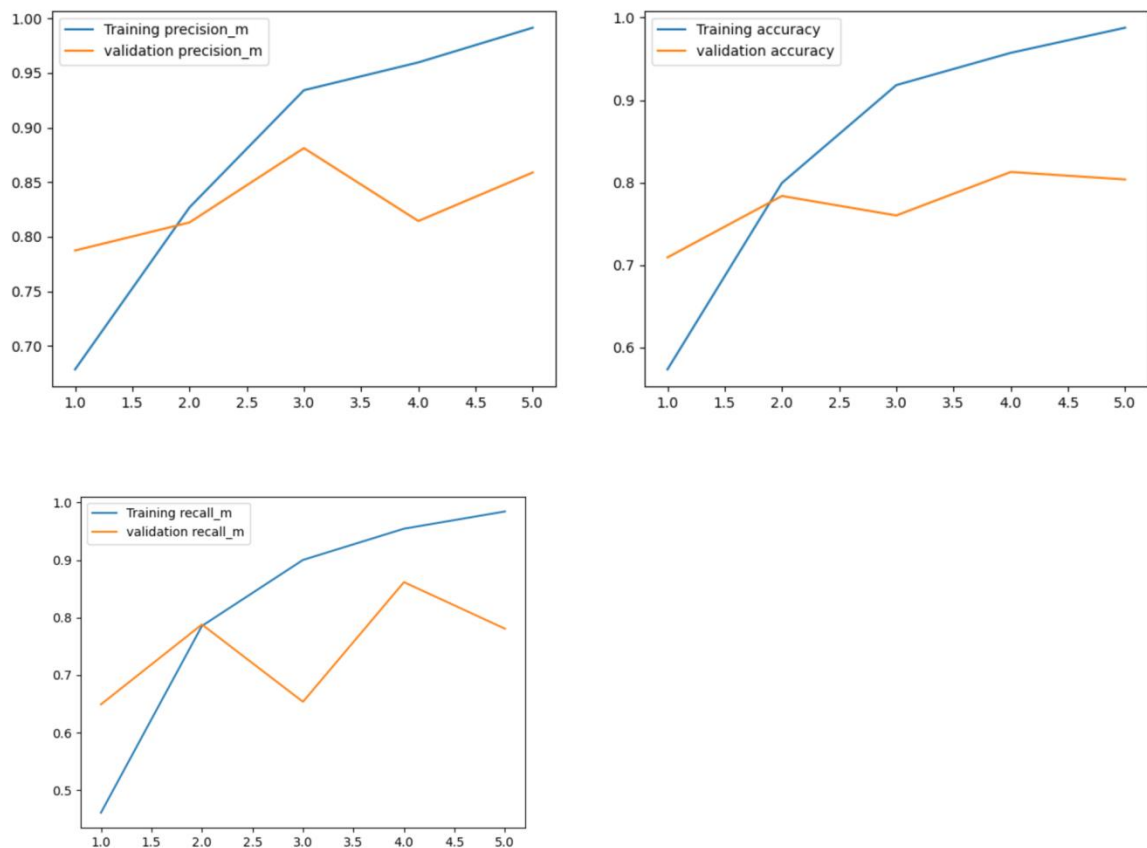
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, None, 64)	874432
lstm_1 (LSTM)	(None, 64)	33024
dense_1 (Dense)	(None, 64)	4160
dense_2 (Dense)	(None, 1)	65
Total params: 911,681		
Trainable params: 911,681		
Non-trainable params: 0		

Σχήμα 8 2η Αρχιτεκτονική μοντέλου RNN

Combination of reviews

Metrics	#1	validation	#2	validation
Precision	0. 5014	0. 5231	0. 9913	0. 8588
Recall	1. 000	1. 000	0. 9843	0. 7805
Accuracy	0. 5014	0. 5164	0. 9877	0. 8036

Σχήμα9 Combination of reviews results with RNN



Σχήμα10 Combination of Reviews RNN Graphs

Για το “σετ” δεδομένων προερχόμενο από Amazon, Yelp και IMDB, υπήρξε μεγάλη διαφορά μεταξύ των δυο νευρωνικών δικτύων, με το δεύτερο να έχει πολλή καλύτερη απόδοση και στα τρία μέτρα σύγκρισης που χρησιμοποιήθηκαν. Εν συνεχεία, η απόδοση του δεύτερου RNN, φαίνεται να είναι μην διαφέρει από τις καλύτερες αποδόσεις του RF(Random Forest)

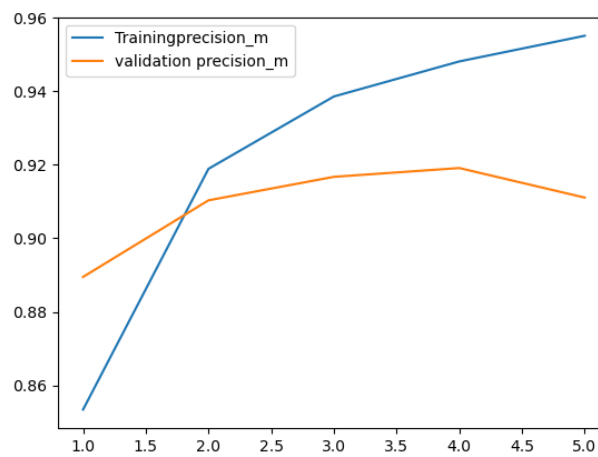
RF	RNN
0. 88	0. 8588
0. 73	0. 7805
0. 80	0. 8036

Σχήμα11 RF vs RNN for Combinations of reviews

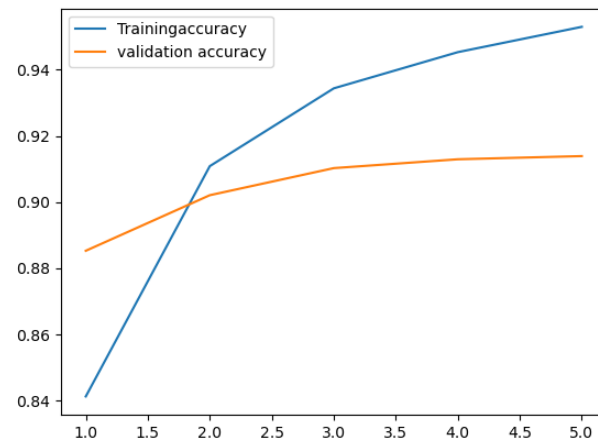
Stanford Dataset

Metrics	#1	Validation	#2	Validation
Precision	1	1	0.955	0.911
Recall	0.006	0.002	0.959	0.9336
Accuracy	0.79	0.78	0.953	0.9139

Σχήμα 12 Stanford RNN results



Σχήμα 13 Stanford Precision Graph



Σχήμα 14 Stanford Accuracy Graph

Για το “σετ” δεδομένων από το Stanford, είναι ενδιαφέρον το γεγονός ότι με το πρώτο RNN, το Precision έχει τιμή 1 και το Recall 0, κάτι που ίσως σημαίνει ότι το μοντέλο ταξινομεί όλα τα παραδείγματα σε μια κατηγορία. Με το μεγαλύτερο RNN, οι τιμές είναι πιο φυσιολογικές και συγκρίνοντάς τις με τα αποτελέσματα του RF, υπάρχει αρκετή βελτίωση,

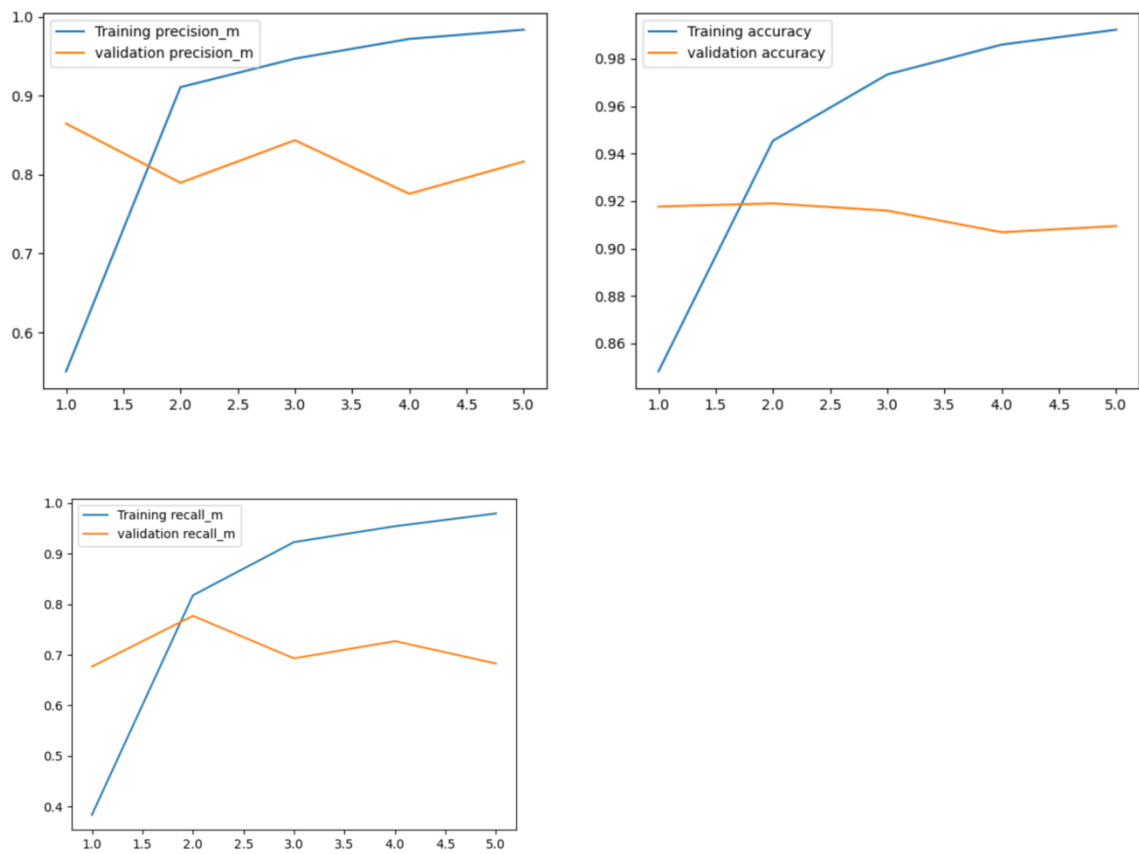
RF	RNN
0. 89	0. 91
0. 94	0. 93
0. 90	0. 91

Σχήμα15 RF vs RNN with Stanford dataset

Twitter Airline Dataset

Metrics	#1	Validation	#2	Validation
Precision	0. 2013	0. 2153	0. 9839	0. 8167
Recall	1	0. 9863	0. 9794	0. 6828
Accuracy	0. 2015	0. 2178	0. 9922	0. 9095

Σχήμα16 Twitter Airline Dataset results with RNN



Σχήμα17 Twitter Airline Dataset RNN graphs

4. 4 Ανάπτυξη του LDA

Ως σώμα κειμένου για το τελικό μοντέλο χρησιμοποιήθηκαν γύρω στα 2000 (δύο χιλιάδες) άρθρα, κυρίως πολιτικού και τεχνολογικού περιεχόμενου. Τα άρθρα αυτά, προήλθαν από τις ιστοσελίδες www.bbc.com και www.aljazeera.com.

Αρχικά, έγιναν κάποιες δοκιμές με διαφορετικό αριθμό κειμένων και διαφορετικό αριθμό θεμάτων. Ως μέτρα χρησιμοποιήθηκαν το Perplexity και το U-mass coherence (McCallum, 2011). Το Perplexity ορίζεται ως

$$b^{-\frac{1}{N} \sum_{i=1}^N \log_b q(x_i)}$$

Όπου q είναι το μοντέλο και x_i ως N ή είσοδος του μοντέλου.

Το U-mass coherence δημιουργήθηκε από τους (McCallum, David Mimno-Edmund Talley-Miriam Leenders-Hanna M. Wallah-Andrew, 2011).

Για το perplexity είναι γνωστό πως όσο μειώνεται, σημαίνει ότι το μοντέλο βελτιώνεται, ενώ για το U-mass coherence, βελτιώνεται όσο πλησιάζει στο μηδέν.

Παρακάτω βρίσκονται τα αποτελέσματα:

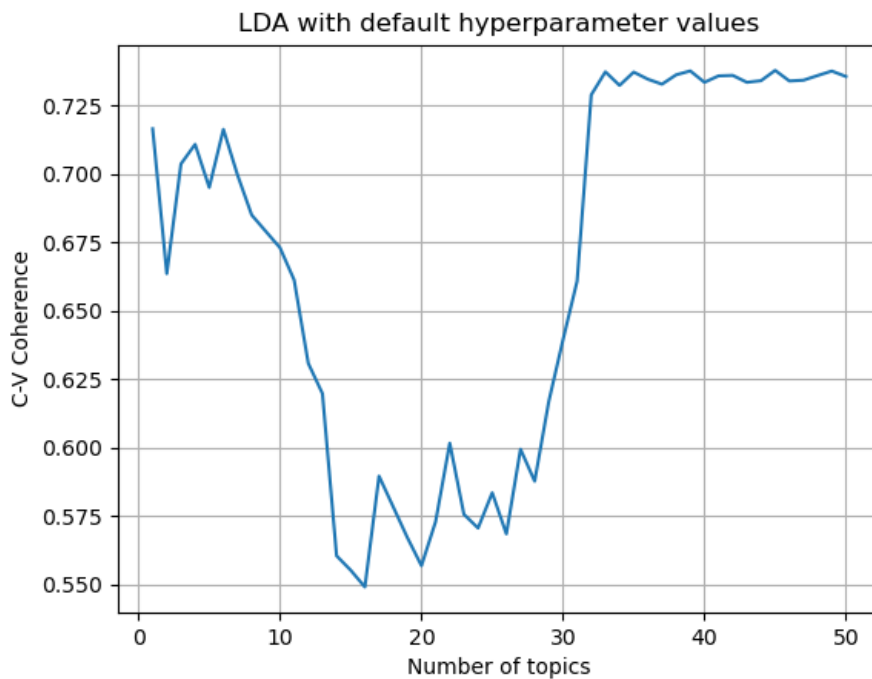
Model	Perplexity	U-mass	Articles	Topics
LDA10	-8. 27	-9. 366	500	10
LDA20	-13. 652	-8. 049	500	20
lda_10_1835	-8. 488	-5. 999	1835	10
lda_20_1835	-12. 778	-8. 678	1835	20

Σχήμα 18 LDA model comparison

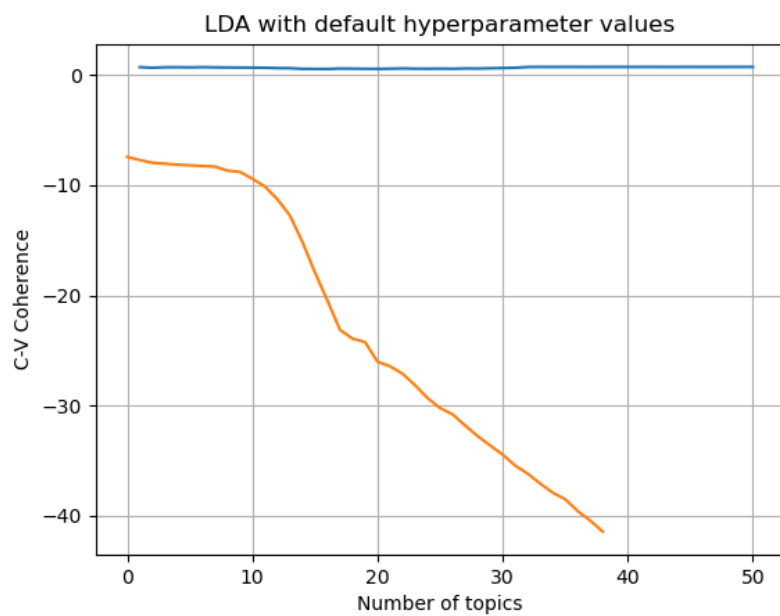
Στη συνέχεια δοκιμάστηκαν κάποια μοντέλα με τους τίτλους 2450 άρθρων ως σώμα κειμένων. Επειδή οι τιμές του U-mass coherence δεν έδειχναν να βοηθούν στη διάκριση μεταξύ καλύτερων και χειρότερων μοντέλων προστέθηκε το CV-coherence.

Model	Perplexity	U-mass Coherence	CV Coherence	Topics
lda_2450	-8. 128	-19. 412	0. 694	5
lda10_2450	-8. 785	-19. 537	0. 672	10
lda15_2450	-15. 156	-18	0. 555	15
lda_20_2450	-24. 223	-17. 727	0. 556	20
lda30_2450	-33. 610	-18. 760	0. 639	30
lda40_2450	-50. 990	-19. 720	0. 735	40

Σχήμα19 LDA models with all metrics



Σχήμα20 CV-Coherence by number of topics



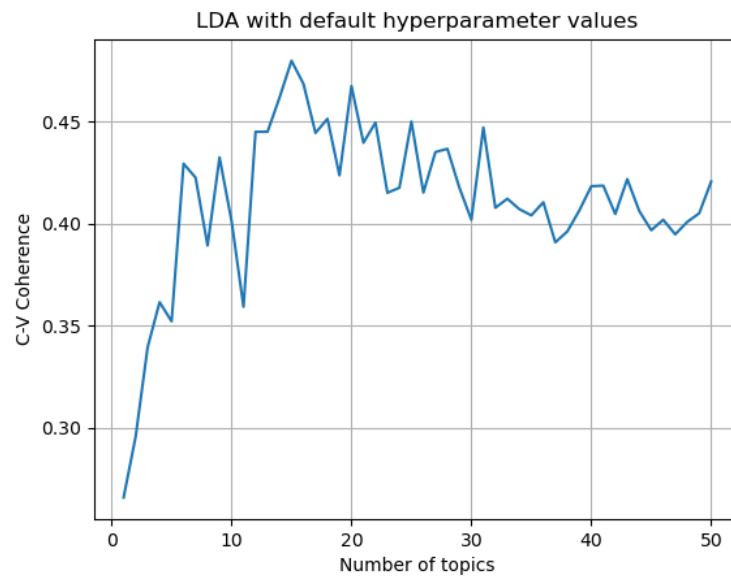
Σχήμα21 CV-Coherence and Perplexity along number of topics

Έπειτα, έγινε μια πιο εκτενής δοκιμή αριθμού θεμάτων, με σώμα κειμένου το σύνολο των άρθρων.

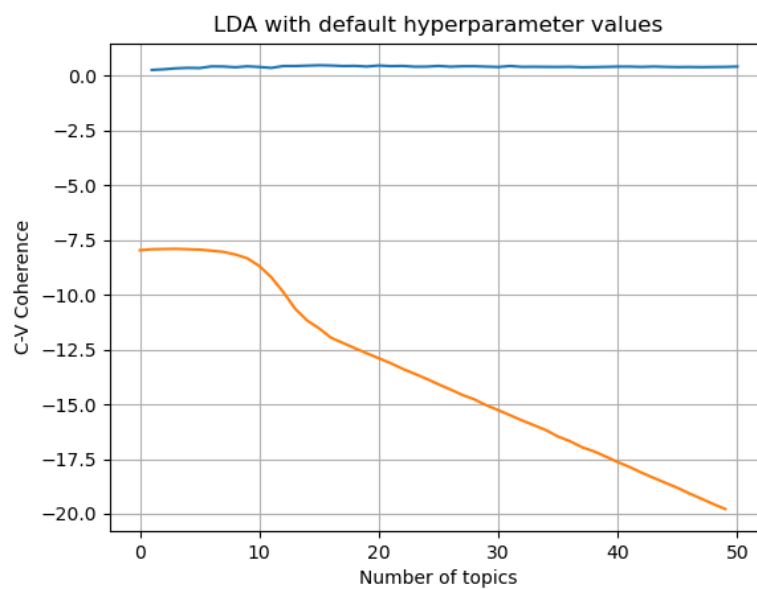
Model	Perplexity	CV Coherence	Topics
lda_2450	-7. 921	0. 352	5
lda10_2450	-8. 337	0. 401	10
lda15_2450	-11. 174	0. 479	15
lda_20_2450	-12. 674	0. 467	20
lda30_2450	-15. 047	0. 401	30
lda40_2450	-17. 387	0. 418	40
lda50_2450	-19. 784	0. 420	50
lda55_2450	-21. 022	0. 406	55
lda60_2450	-22. 208	0. 396	60
lda70_2450	-24. 631	0. 349	70
lda100_2450	-31. 802	0. 352	100

Σχήμα22 LDA models for different number of topics

Φαίνεται, ότι το cv_coherence μεγιστοποιείται όταν τα θέματα είναι 15, η διαφορά όμως, είναι μικρή. Το perplexity δεν έχει όσο χαμηλές τιμές είχε στα προηγούμενα μοντέλα και φαίνεται να μειώνεται συνεχώς με την αύξηση των θεμάτων.



Σχήμα 23 C-V-Coherence graph



Σχήμα 24 LDA model Perplexity and Coherence combined Graph

Για να γίνει πιο κατανοητή η διαφορά μεταξύ των μοντέλων, έγινε μια δοκιμή χρησιμοποιώντας δύο μοντέλα με το ίδιο σώμα κειμένων. Το πρώτο μοντέλο διαχώρισε τα κείμενα σε 15 θέματα και το δεύτερο σε 50.

Δόθηκε 1 άρθρο και στη συνέχεια υπολογίστηκαν σχετικά άρθρα, βάσει των δύο μοντέλων. Οι απαντήσεις των μοντέλων κατηγοριοποιήθηκαν σε irrelevant, slightly relevant και relevant.

Απόφο: School attendance back at high levels in England - BBC News

MODEL 1

1. "We were the only people there to support them" - BBC News **irrelevant**
2. "Over 50 and overlooked for work" - BBC News **irrelevant**
3. 'Covid and cancer: 'It felt like the universe was out to get me' - BBC News slightly **relevant**
4. Nurseries sent first official cyber-attack warning - BBC News **irrelevant**
5. "Coronavirus: 'We got through lockdown by dancing' - BBC News" **relevant**
6. Online privacy: digitally exposed | Science and Technology News | Al Jazeera **irrelevant**
7. Covid in Wales: More face-to-face university teaching after Easter - BBC News **relevant**
8. Covid-19: Lateral flow testing for years 12-14 pupils - BBC News **relevant**
9. Watch: Freestyle Ski and Snowboarding World Championships - Snowboard Big Air finals - BBC Sport **irrelevant**
10. Covid: Wales' school pupils 'excited and nervous' about return - BBC News **relevant**

MODEL 2

1. Coronavirus: 'Everyone is excited to be back at school' - BBC News **relevant**
2. Nurseries sent first official cyber-attack warning - BBC News **irrelevant**
3. LIVE: Scottish Conservative Party conference speech by Douglas Ross - BBC News **irrelevant**
4. Covid in Wales: More face-to-face university teaching after Easter - BBC News **relevant**
5. Censors hold back Chinese cinema | News News | Al Jazeera **irrelevant**
6. Guernsey's autism centre to move schools over space issue - BBC News **irrelevant**
7. Covid: Outdoor education centres 'may not survive' - BBC News **relevant**
8. Covid-19: Lateral flow testing for years 12-14 pupils - BBC News **relevant**
9. Ipswich aims to become 'UK's first 15-minute' town - BBC News **irrelevant**
10. Cross-border workers call for home-working tax law - BBC News **irrelevant**

To 1^ο μοντέλο είχε 5 **irrelevant**, 1 slightly **relevant**, 4 **relevant**

To 2^ο μοντέλο είχε 6 irrelevant, 0 slightly relevant, 4 **relevant**

Άρθρο: Covid-19 in the UK: How many coronavirus cases are there in your area? - BBC News

MODEL 1

1. Newcastle city centre £50m revamp plan - BBC News **irrelevant**
2. Covid: Guernsey leaves second pandemic lockdown - BBC News' **relevant**
3. Gold rush threat to Amazon jungle | News News | Al Jazeera **irrelevant**
4. Machine finds tantalising hints of new physics - BBC News **irrelevant**
5. Seen from the sky: Polluted waters around the world | Gallery News | Al Jazeera **irrelevant**
6. COVID jabs bring relief for vulnerable California farmworkers | Agriculture News | Al Jazeera **relevant**
7. Dating agencies boom in China | Arts and Culture News | Al Jazeera **irrelevant**
8. Jinxed Japanese space programme | News News | Al Jazeera **irrelevant**
9. Munching your way through the afternoon | News News | Al Jazeera **irrelevant**
10. Hot tub accidents triple in lockdown, says insurer - BBC News slightly **relevant**

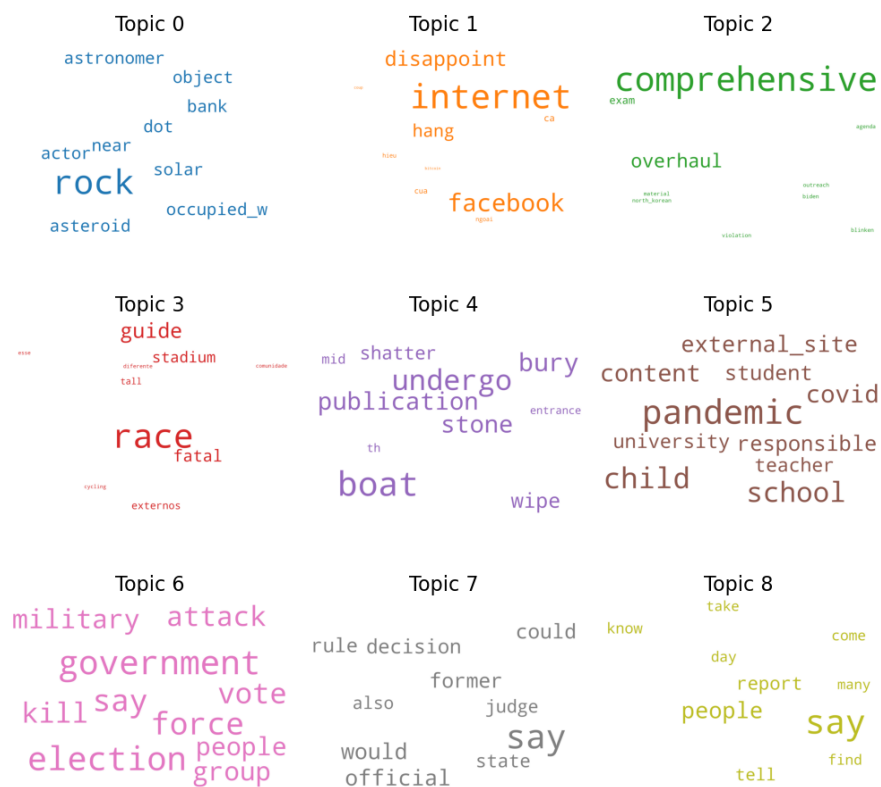
MODEL 2

1. Canada quarantines 500 suspected SARS cases | News News | Al Jazeera slightly **relevant**
2. Covid: Five more in hospital on Isle of Man as 45 new cases confirmed - BBC News **relevant**
3. How the COVID-19 pandemic is affecting mental health | News News | Al Jazeera **relevant**
4. Japan is greying fast | News News | Al Jazeera **irrelevant**
5. Covid: 17 patients treated after Ysbyty Gwynedd outbreak - BBC News' **relevant**
6. 'Covid-19: Brazil to get fourth health minister since pandemic began - BBC News **relevant**
7. Why is India staring at a 'second peak' of COVID cases? | Coronavirus pandemic News | Al Jazeera **relevant**
8. Covid-19: India reports record daily rise in new infections - BBC News **relevant**
9. Russia identifies two cases of South African COVID variant | Coronavirus pandemic News | Al Jazeera **relevant**

10. COVID jabs bring relief for vulnerable California farmworkers | Agriculture News |
Al Jazeera **relevant**

To 1^ο μοντέλο είχε 7 irrelevant, 1 slightly relevant, 2 **relevant**

To 2^ο μοντέλο είχε 1 irrelevant, 2 slightly relevant, 7 **relevant**

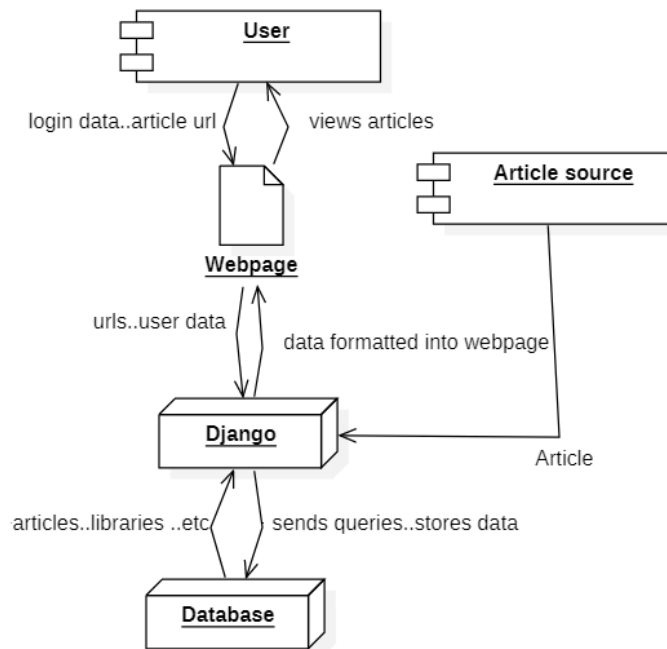


Σχήμα 25 LDA Topic Visualization

Στο σχήμα 26 απεικονίζονται ορισμένες λέξεις, για κάθε θέμα του μοντέλου 1.

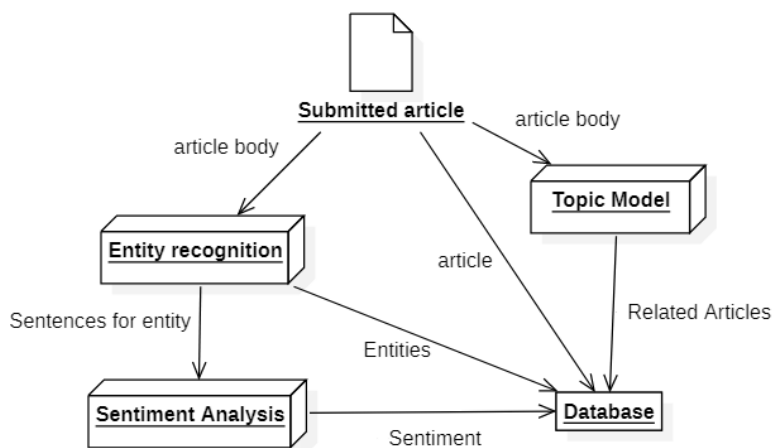
Κεφάλαιο 5 Η εφαρμογή

5.1 Βάση δεδομένων



Σχήμα 26 Διάγραμμα Ροής Δεδομένων εφαρμογής

Στο σχήμα 27 παρουσιάζονται οι διάφορες ροές δεδομένων της εφαρμογής. Ο χρήστης βλέπει το περιεχόμενο της εφαρμογής είτε είναι άρθρα, βιβλιοθήκες ή δεδομένα. Επίσης ο χρήστης καταχωρεί δεδομένα στην εφαρμογή, τα οποία περνάνε στο Django. Όταν καταχωρεί σύνδεσμο άρθρου, η πληροφορία μεταφέρεται στο Django και από το Django γίνεται συλλογή δεδομένων από την αρχική πηγή του άρθρου. Στη συνέχεια αποθηκεύεται στην βάση. Το Django βέβαια αναλαμβάνει και τη δημιουργία του περιεχομένου που αποστέλλεται στον χρήστη σε μορφή ιστοσελίδας, καλώντας τα δεδομένα από την βάση.

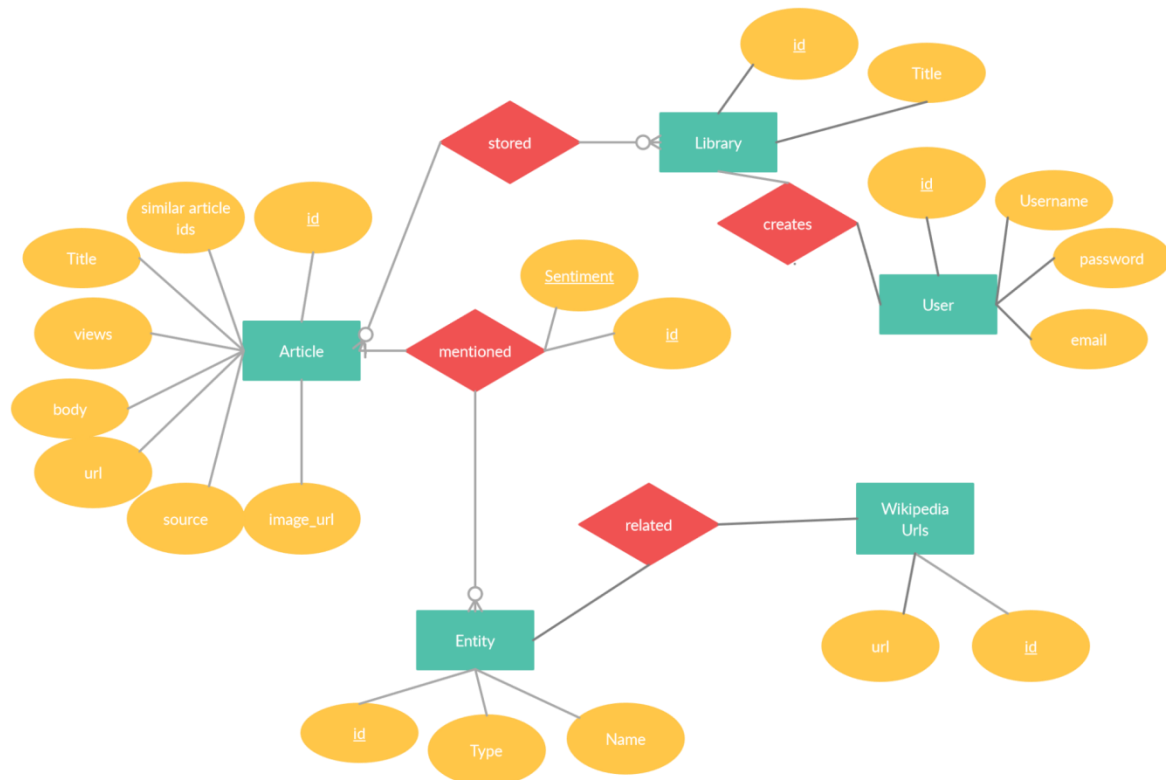


Σχήμα 27 Διαδικασία Υποβολής και ανάλυσης άρθρου

Στο σχήμα 28 περιγράφεται η διαδικασία επεξεργασίας των άρθρων. Αφότου υποβληθεί το άρθρο, αρχικά συμβαίνει η διαδικασία αναγνώρισης οντοτήτων, στην συνέχεια για κάθε οντότητα γίνεται ανάλυση συναισθήματος στις προτάσεις που περιέχουν την οντότητα. Ακόμη γίνεται εύρεση παρόμοιων άρθρων και τα δεδομένα αυτά αποθηκεύονται στην βάση.

Ως βάση δεδομένων χρησιμοποιήθηκε η Postgresql. Οι πίνακες δημιουργήθηκαν μέσω του συστήματος Object Relational Mapper (ORM), που περιέχει το Django. Το ORM δίνει τη δυνατότητα να δημιουργούνται και να επεξεργάζονται πίνακες σε γλώσσα Python. Ακόμη, για τον πίνακα του χρήστη χρησιμοποιήθηκε το σύστημα αυθεντικοποίησης χρήστη, του Django (django.contrib.auth).

Ο πίνακας του χρήστη περιέχει τα πεδία που προσφέρονται από το σύστημα αυθεντικοποίησης του django, τα οποία είναι id, κωδικός, όνομα χρήστη, διεύθυνση email, ημερομηνία δημιουργίας λογαριασμού, ημερομηνία τελευταίας σύνδεσης, τα πεδία δυαδικής τιμής διαχειριστής, εργαζόμενος και ενεργός.

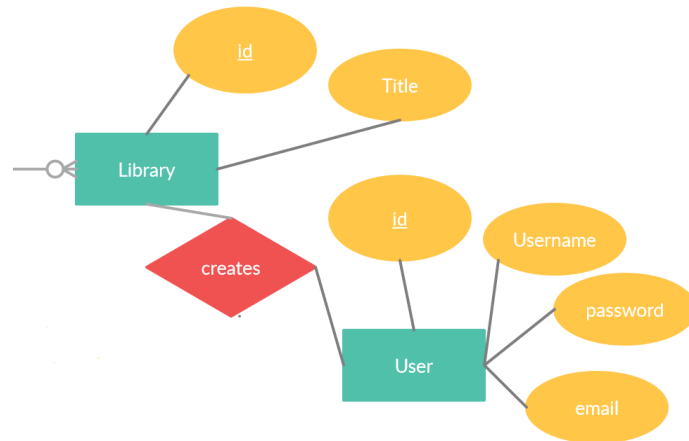


Σχήμα 28 Σχεδιάγραμμα οντοτήτων συσχετίσεων της βάσης δεδομένων

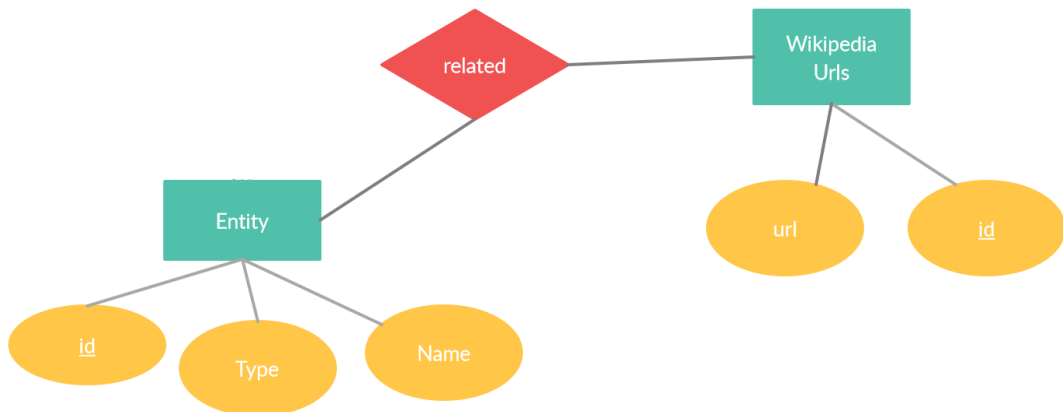
Ο πίνακας Library περιέχει τα πεδία id και τίτλος. Έχει σχέση ένα προς πολλά με τον πίνακα User, καθώς κάθε χρήστης μπορεί να έχει πολλές βιβλιοθήκες. Επίσης, έχει σχέση μια πολλά προς πολλά με τον πίνακα Article, καθώς κάθε άρθρο μπορεί να καταχωρηθεί σε πολλές βιβλιοθήκες.

Ο πίνακας του άρθρου αποτελείται από τα πεδία τίτλος, σώμα, πηγή, σύνδεσμος, σύνδεσμος εικόνας, ημερομηνία αποθήκευσης, προβολές και σχετιζόμενα άρθρα. Οι περισσότεροι πίνακες περιέχουν και ένα πεδίο id.

Στον πίνακα Entity, καταχωρούνται τα πρόσωπα, περιοχές και οργανισμοί που αναγνωρίζονται εντός των άρθρων. Έχει πεδία όνομα και τύπος και έχει σχέση πολλά προς πολλά με τον πίνακα άρθρο.



Σχήμα 29 Σχεδιάγραμμα οντοτήτων συσχετίσεων Χρήστη και Βιβλιοθηκών



Σχήμα 30 Σχεδιάγραμμα οντοτήτων συσχετίσεων οντότητας και συνδέσμων wikipedia

Στο σχήμα Β'.5 απεικονίζεται το σχεδιάγραμμα οντοτήτων συσχετίσεων των οντοτήτων και των συνδέσμων wikipedia.

5. 2 Object Relational Mapper

Παρακάτω ακολουθεί μια επεξήγηση κάποιων μερών του κώδικα του, που χρησιμοποιήθηκε για τη δημιουργία των απαιτούμενων πινάκων.

```
class Entity(models.Model):
    """docstring for Entity."""
    name=models.CharField(max_length=128,unique=True)
    class Ent_Type(models.TextChoices):
        PERSON='PERSON'
        GEOPOLITICAL_ENTITY='GPE'
        ORGANIZATION='ORG'
        GEOGRAPHICAL_ENTITY='GEO'
    type=models.CharField(max_length=10,choices=Ent_Type.choices,default=Ent_Type.PERSON)
    articles = models.ManyToManyField(Property, through='Sentiment')
```

Σχήμα 32

Στο σχήμα 33 εμφανίζεται ο κώδικας για τη δημιουργία του πίνακα των οντοτήτων. Εκτός από τον ορισμό των πεδίων όνομα και άρθρα υπάρχει και μια κλάση με το όνομα Ent_Type.

Η κλάση αυτή χρησιμοποιείται με σκοπό την εξασφάλιση ότι στο πεδίο τύπος θα αποθηκεύονται οντότητες μόνο των επιλογών που φαίνονται εντός της κλάσης.

```
class Article(models.Model):
    name=models.CharField(max_length=500,default='')
    url=models.CharField(max_length=500,default='')
    body=models.CharField(max_length=20000,default='')
    image_file = models.ImageField(max_length=700,upload_to='media', default='default.jpg')
    image_url = models.URLField(max_length=1000,default='www.noimage.com')
    #Here we store for each article the ids of the article similar to it
    similar_ids=ArrayField(models.IntegerField(), default=list,blank=True)
    views=models.IntegerField(default=0)
    date=models.DateField(auto_now_add=True)
    source=models.CharField(max_length=50,default='')
    def __str__(self):
        return self.name
```

Σχήμα 33

Στο σχήμα 34, απεικονίζεται ο ορισμός του πίνακα των άρθρων. Ένα πεδίο που η φύση του είναι λιγότερο εμφανής από τα υπόλοιπα, είναι το similar_ids. Το πεδίο αυτό είναι τύπου ArrayField, το οποίο στο Django δίνει τη δυνατότητα να αποθηκεύονται λίστες και στη συγκεκριμένη περίπτωση, λίστες ακέραιων αριθμών.

Αυτό το πεδίο χρησιμοποιήθηκε ώστε να αποθηκεύονται τα αναγνωριστικά άρθρων που έχουν υπολογιστεί μέσω του μοντέλου LDA ως παρόμοια με το άρθρο. Μέσω αυτού το

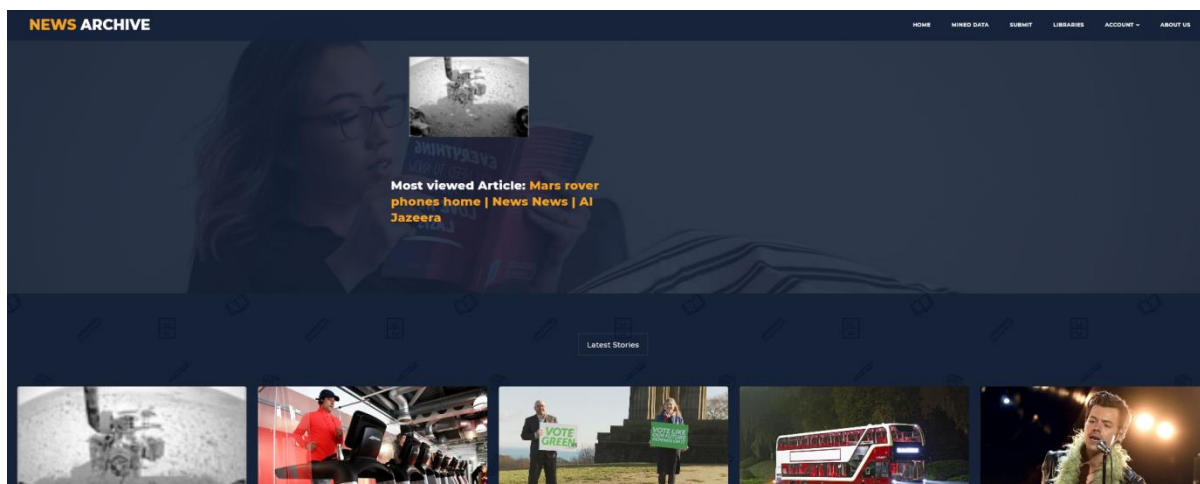
πεδίου, επιτυγχάνεται μια απλούστερη αρχιτεκτονική από μια ενδεχομένως πιο καθιερωμένη λύση αυτού του προβλήματος, όπου ο πίνακας άρθρο θα ήταν συσχετιζόμενος με τον εαυτό του.

```
class Sentiment(models.Model):
    """docstring for Sentiment."""
    article=models.ForeignKey(Property, on_delete=models.CASCADE)
    entity=models.ForeignKey(Entity, on_delete=models.CASCADE)
    sentiment=models.BooleanField()
    class Meta:
        constraints=[
            models.UniqueConstraint(fields=['article', 'entity'], name='only_one_result_for_article -entity')
        ]
```

Σχήμα 34

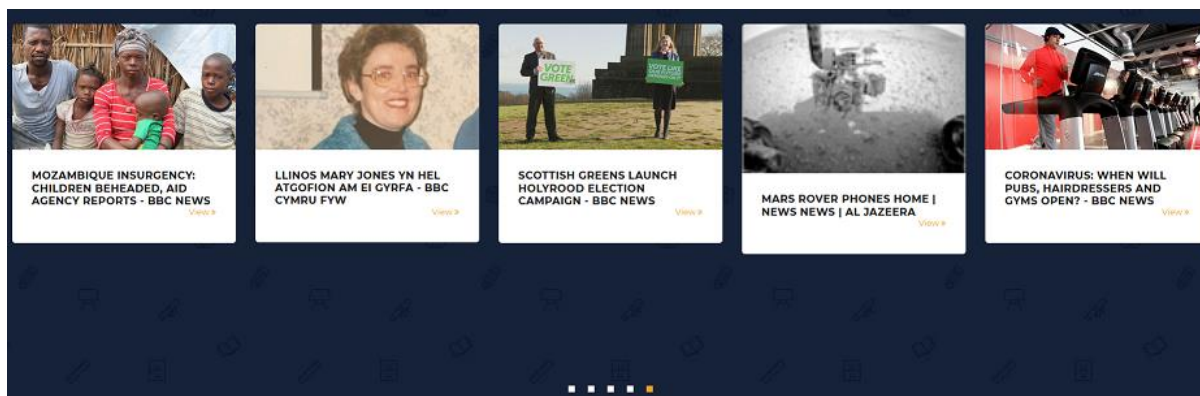
Στον συγκεκριμένο ορισμό του πίνακα Sentiment υπάρχει μια κλάση Meta, όπου ορίζεται μια τιμή constraints. Η τιμή αυτή περιέχει έναν περιορισμό που εμποδίζει να υπάρχει παραπάνω από μια φορά η αντιστοίχιση ενός άρθρου με μία οντότητα. Αυτό γίνεται ώστε να αποφευχθεί να αποθηκεύονται πολλαπλές φορές οι ίδιες πληροφορίες.

5.3 Αρχική Σελίδα



Σχήμα 35

Παραπάνω βρίσκεται η αρχική σελίδα της εφαρμογής. Ξεκινώντας από πάνω προς τα κάτω, υπάρχει ο τίτλος της ιστοσελίδας και στα δεξιά, το βασικό μενού. Στη συνέχεια, εμφανίζεται το άρθρο με τις περισσότερες προβολές και παρακάτω, τα τελευταία άρθρα που έχουν υποβληθεί στην ιστοσελίδα. Ο χρήστης έχει τη δυνατότητα να διαβάσει το άρθρο πατώντας τον τίτλο ή να διαβάσει κάποιο από τα τελευταία άρθρα πατώντας στο πλαίσιο του άρθρου.



Σχήμα 36

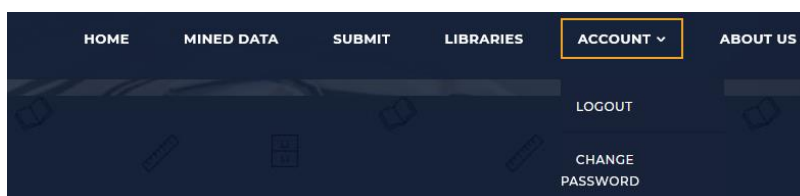
Τα τελευταία άρθρα προβάλλονται σε σειρά σε ένα πλαίσιο το οποίο κινείται και ο χρήστης μπορεί είτε να σύρει το ποντίκι, είτε να επιλέξει κουτάκι από κάτω τα άρθρα, ώστε να τα προσπελάσει.



Σχήμα 37

Το παραπάνω μενού βρίσκεται στην κορυφή κάθε σελίδας του ιστότοπου.

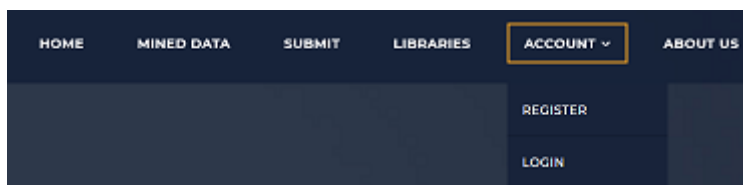
Η επιλογή HOME, μεταφέρει τον χρήστη στην αρχική σελίδα, η επιλογή MINED DATA μεταφέρει τον χρήστη σε μια σελίδα όπου προβάλλονται ενδιαφέροντα στατιστικά για τα άρθρα που έχουν υποβληθεί στην εφαρμογή. Η επιλογή SUBMIT μεταφέρει τον χρήστη στη σελίδα υποβολής άρθρων. Πατώντας την επιλογή LIBRARIES, ο χρήστης μεταφέρεται στη σελίδα όπου εμφανίζονται οι βιβλιοθήκες που έχει δημιουργήσει.



Σχήμα 38

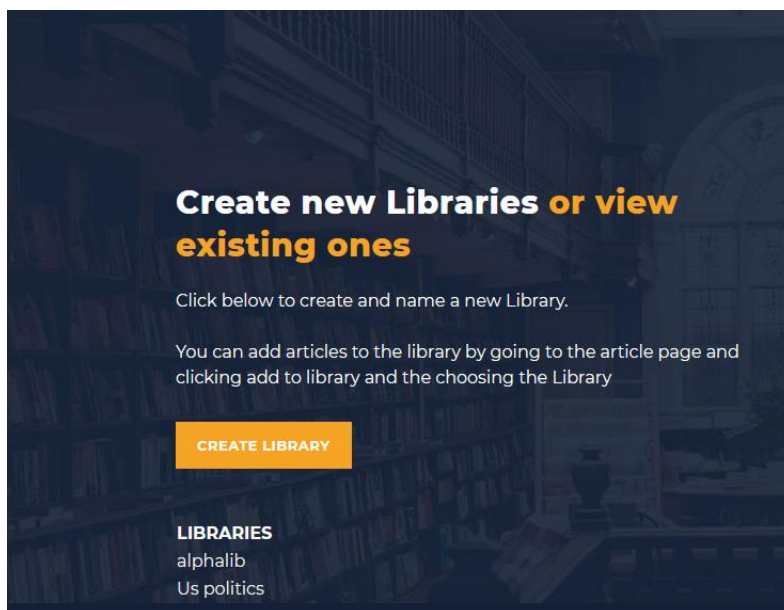
Επιλέγοντας το ACCOUNT, ο χρήστης έχει τη δυνατότητα είτε να προχωρήσει στην σελίδα αλλαγής κωδικού, είτε να αποσυνδεθεί από τον λογαριασμό του.

Εάν ο χρήστης δεν είναι συνδεδεμένος, επιλέγοντας το ACCOUNT θα έχει την επιλογή είτε να δημιουργήσει νέο λογαριασμό, είτε να συνδεθεί σε υπάρχοντα λογαριασμό, όπως φαίνεται στην εικόνα 40.



Σχήμα 39

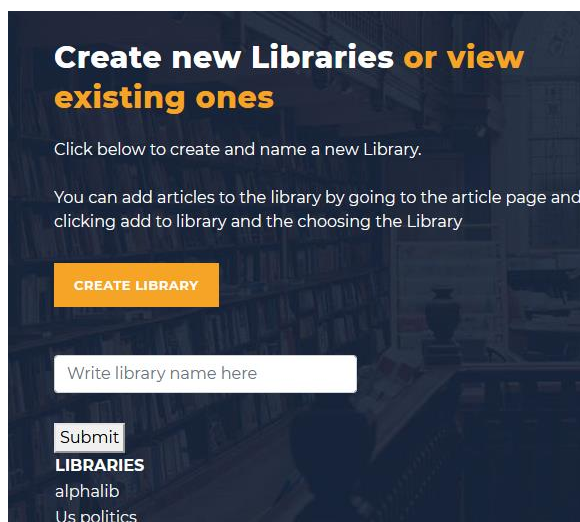
5. 4 Βιβλιοθήκες



Σχήμα 40

Στο σχήμα 41 προβάλλεται η σελίδα των βιβλιοθηκών. Ο χρήστης εφόσον είναι συνδεδεμένος, μπορεί να πατήσει το κουμπί δημιουργία βιβλιοθήκης. Τότε, εμφανίζεται ένα πλαίσιο για την ονομασία της βιβλιοθήκης και αφότου έχει δοθεί ο τίτλος της βιβλιοθήκης, το τελευταίο βήμα είναι να πατηθεί το πλήκτρο Submit. Το πλαίσιο και το κουμπί, προβάλλονται στο σχήμα 42.

Κάτω από το κουμπί “δημιουργία βιβλιοθήκης” βρίσκεται η λίστα ήδη υπαρχόντων βιβλιοθηκών. Εφόσον ο χρήστης επιλέξει μια από τις βιβλιοθήκες, μεταφέρεται στη σελίδα της βιβλιοθήκης όπου εμφανίζεται η λίστα των άρθρων που έχουν καταχωρηθεί σε αυτή.



Create new Libraries or view existing ones

Click below to create and name a new Library.

You can add articles to the library by going to the article page and clicking add to library and then choosing the Library

CREATE LIBRARY

Write library name here

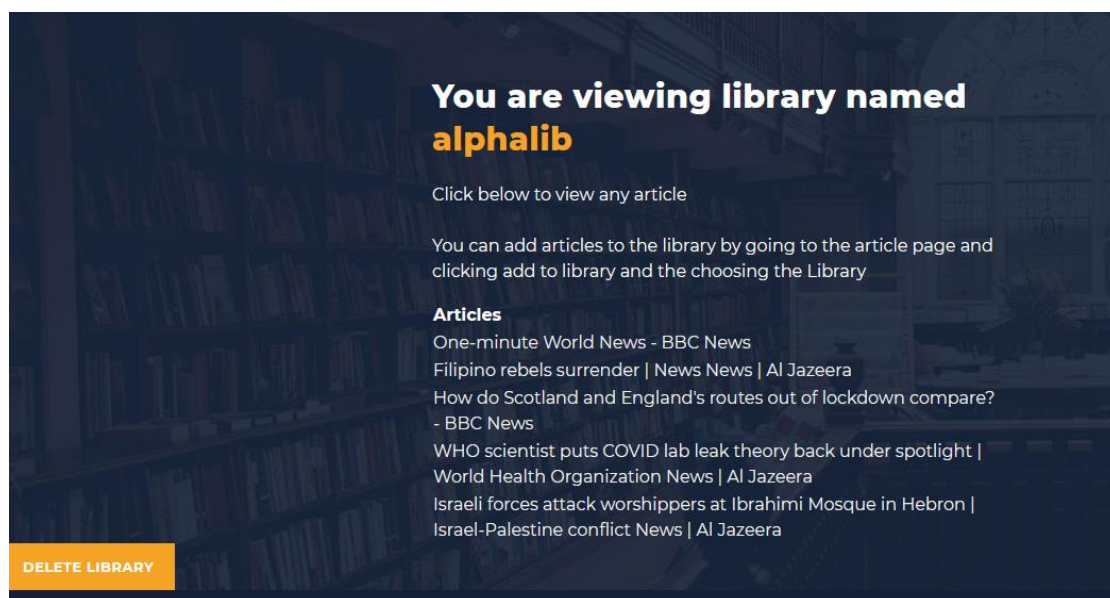
Submit

LIBRARIES

alphalib

Us politics

Σχήμα 41



You are viewing library named alphalib

Click below to view any article

You can add articles to the library by going to the article page and clicking add to library and then choosing the Library

Articles

One-minute World News - BBC News

Filipino rebels surrender | News News | Al Jazeera

How do Scotland and England's routes out of lockdown compare? - BBC News

WHO scientist puts COVID lab leak theory back under spotlight | World Health Organization News | Al Jazeera

Israeli forces attack worshippers at Ibrahimi Mosque in Hebron | Israel-Palestine conflict News | Al Jazeera

DELETE LIBRARY

Σχήμα 42

Στο σχήμα 43 προβάλλεται η σελίδα μιας βιβλιοθήκης που ονομάζεται alphalib. Κάτω από τον τίτλο Articles, εμφανίζεται η λίστα των τίτλων των άρθρων που έχουν καταχωρηθεί στη βιβλιοθήκη. Οι τίτλοι είναι συγχρόνως και σύνδεσμοι που μεταφέρουν τον χρήστη στη σελίδα του άρθρου.

Κάτω από τα άρθρα εμφανίζεται το κουμπί διαγραφή βιβλιοθήκης, το οποίο βεβαίως διατηρεί τα άρθρα στη βάση, αλλά διαγράφει τη βιβλιοθήκη του χρήστη.

5.5 Άρθρο

Η σελίδα του άρθρου έχει στην κορυφή της τον τίτλο του άρθρου και στη συνέχεια, το κείμενο (δείτε σχήμα 44). Από κάτω, παρέχεται ο σύνδεσμος από όπου προήλθε το άρθρο και στα δεξιά του κειμένου προβάλλεται μια εικόνα που έχει καταχωρηθεί ως εικόνα σχετιζόμενη με το άρθρο, κατά την υποβολή του άρθρου. Ακόμη, από κάτω από το άρθρο εμφανίζεται το κουμπί ADD TO LIBRARY (προσθήκη σε βιβλιοθήκη). Εφόσον ο χρήστης το πατήσει, εμφανίζονται οι βιβλιοθήκες του χρήστη, με τον ίδιο τρόπο που προβάλλεται ακολούθως στο σχήμα 45. Πατώντας μια από τις βιβλιοθήκες, το άρθρο καταχωρείται στη συγκεκριμένη βιβλιοθήκη.

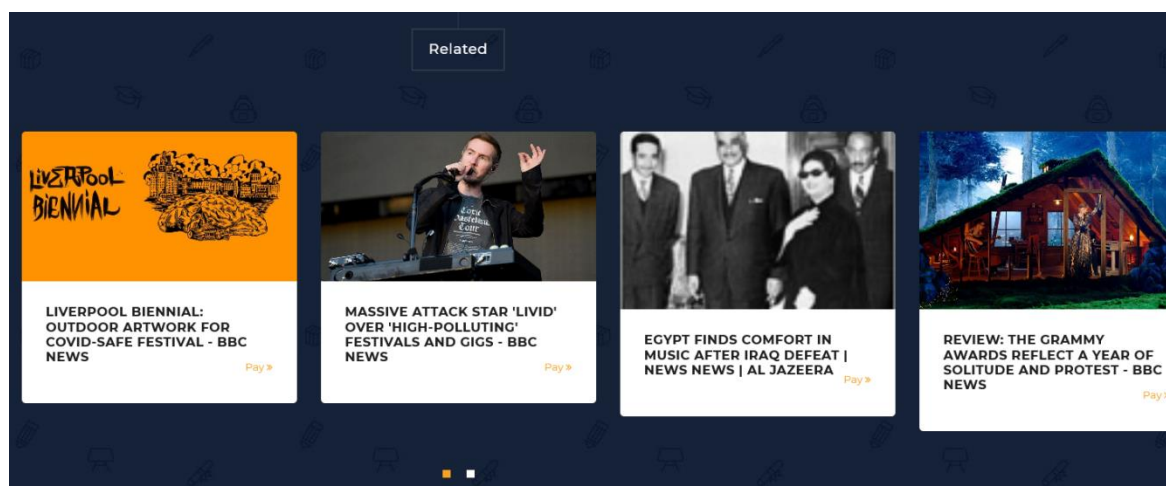


Σχήμα 43



Σχήμα 44

Κάτω από το άρθρο προβάλλονται τα σχετιζόμενα άρθρα. Η λίστα των σχετιζόμενων άρθρων για κάθε άρθρο έχει δημιουργηθεί μέσω του μοντέλου LDA, για το οποίο έχει γίνει ήδη αναφορά, στο κεφάλαιο 2.



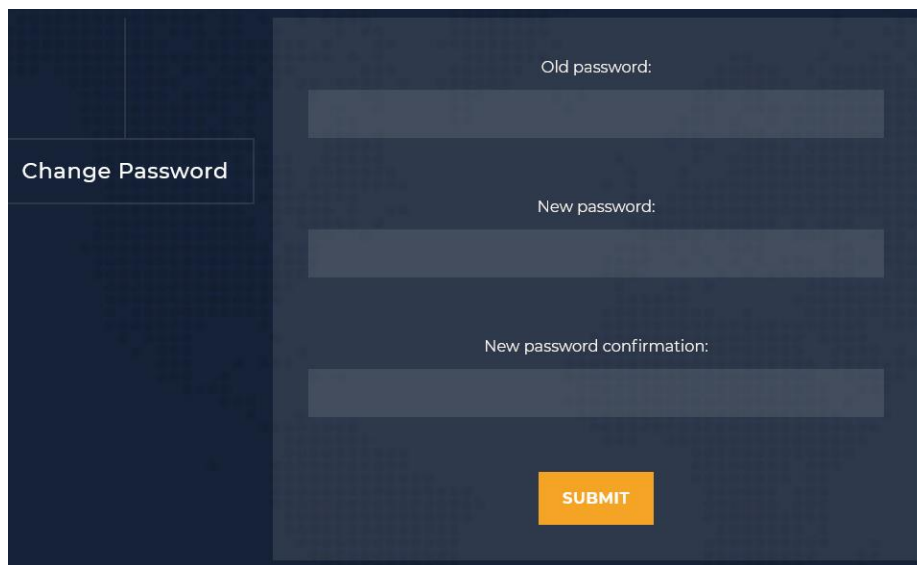
Σχήμα 45

5. 6 Χρήστες

Υπάρχουν τρεις σελίδες σχετιζόμενες με τους λογαριασμούς των χρηστών. Αρχικά, υπάρχει σελίδα εγγραφής νέου χρήστη, σελίδα σύνδεσης χρήστη και σελίδα αλλαγής κωδικού. Οι σελίδες αυτές είναι προσβάσιμες μέσω του μενού, όπως είχε αναφερθεί και νωρίτερα στην ενότητα 5.3.

Σχήμα 46

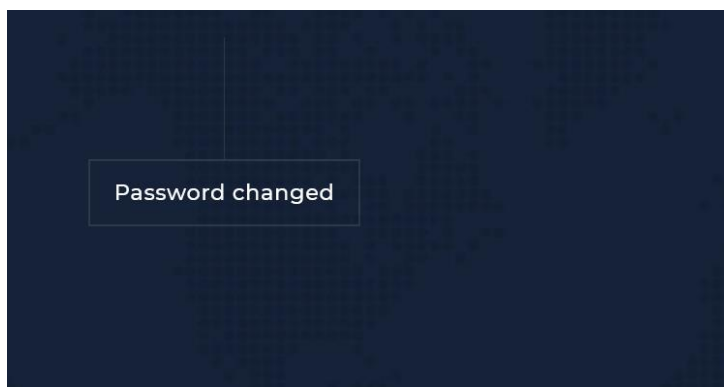
Στο σχήμα 47 εμφανίζεται η φόρμα εγγραφής νέου χρήστη στην εφαρμογή. Τα πεδία που καλείται να συμπληρώσει ο χρήστης είναι: όνομα χρήστη, διεύθυνση email και κωδικός, όπου χρειάζεται και η επαλήθευσή του, όπως συνηθίζεται στις περισσότερες φόρμες εγγραφής.



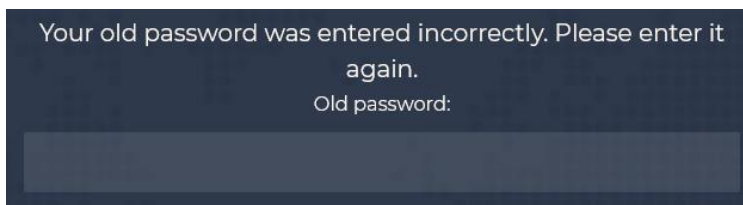
Σχήμα 47

Στο σχήμα 48, εμφανίζεται η φόρμα αλλαγής κωδικού όπου ο χρήστης πρέπει αρχικά να συμπληρώσει τον υπάρχοντα κωδικό και στη συνέχεια τον νέο.

Εφόσον ο χρήστης συμπληρώσει σωστά τον υπάρχοντα κωδικό και τον νέο, μεταφέρεται στη σελίδα επιβεβαίωση αλλαγής κωδικού, που εμφανίζεται στο σχήμα 49. Εάν γίνει λάθος κατά την καταχώρηση του υπάρχοντος κωδικού ή του νέου, εμφανίζεται είτε το μήνυμα λάθους, όπως φαίνεται και στο σχήμα 50, είτε το μήνυμα λάθους, όπως φαίνεται στο σχήμα 51 αντιστοίχως.



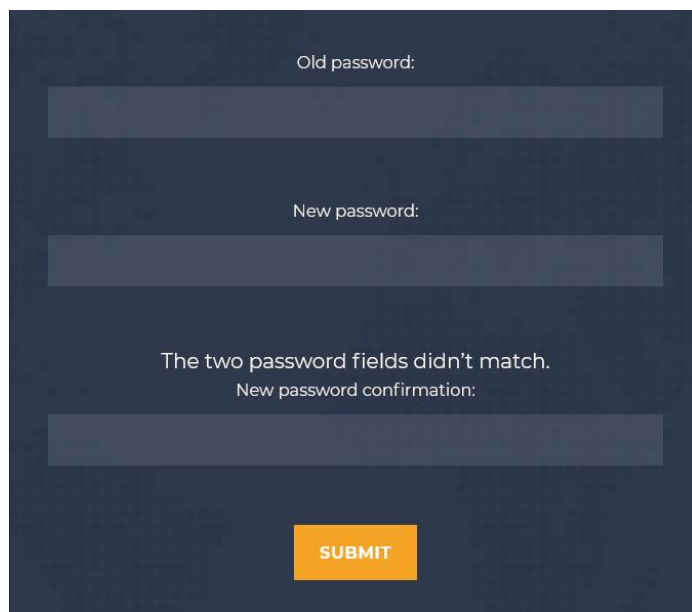
Σχήμα 48



Your old password was entered incorrectly. Please enter it again.

Old password:

Σχήμα 49



Old password:

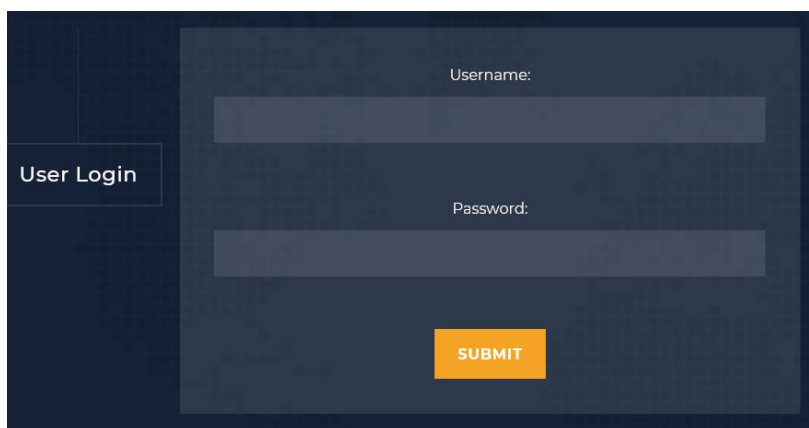
New password:

The two password fields didn't match.

New password confirmation:

SUBMIT

Σχήμα 50



User Login

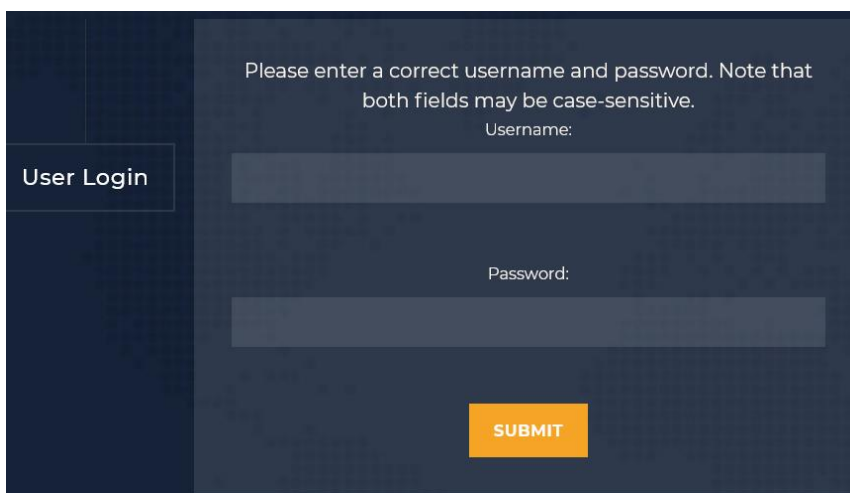
Username:

Password:

SUBMIT

Σχήμα 51

Στο σχήμα 52, εμφανίζεται η φόρμα σύνδεσης χρήστη. Σε περίπτωση καταχώρησης λανθασμένων πληροφοριών, εμφανίζεται το μήνυμα λάθους που απεικονίζεται στο σχήμα 53. Εφόσον ο χρήστης συμπληρώσει τις σωστές πληροφορίες, μεταφέρεται στη κύρια σελίδα.



Please enter a correct username and password. Note that both fields may be case-sensitive.

Username:

Password:

SUBMIT

User Login

Σχήμα 52

5. 7 Σελίδα υποβολής άρθρων

Η σελίδα υποβολής άρθρων είναι η σελίδα, μέσα από την οποία ο χρήστης μπορεί να καταχωρήσει νέα άρθρα στο αρχείο της εφαρμογής. Στο σχήμα 54 απεικονίζεται η πρώτη φόρμα που καλείται να συμπληρώσει ο χρήστης. Σε αυτή τη φόρμα πρέπει να δοθεί ο σύνδεσμος για το άρθρο που επιθυμεί ο χρήστης να καταχωρηθεί.

Αφότου υποβληθεί ο σύνδεσμος, εμφανίζεται μια φόρμα που απεικονίζεται στις εικόνες 55 και 56. Σε αυτή τη φόρμα εμφανίζονται όλα τα δεδομένα του άρθρου. Ο χρήστης έχει τη δυνατότητα να επεξεργαστεί τα δεδομένα αυτά, εφόσον είναι λανθασμένα. Λανθασμένα θεωρούνται τα δεδομένα εκείνα, που δεν σχετίζονται με το άρθρο που ήθελε αρχικά να υποβάλει ο χρήστης, λ.χ. ένα κείμενο που δεν είναι μέρος του άρθρου.

Αφότου ο χρήστης επεξεργαστεί τα δεδομένα ή επιβεβαιώσει ότι τα δεδομένα είναι σωστά, καλείται να πατήσει το κουμπί υποβολής (SUBMIT) .

Σχήμα 53

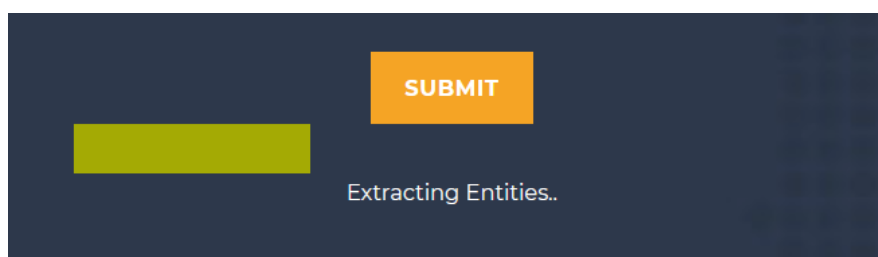
Σχήμα 54

Αφότου υποβληθεί το άρθρο, ξεκινά η επεξεργασία του και εμφανίζεται μια μπάρα, που αναπαριστά την πρόοδο της επεξεργασίας. Η μπάρα απεικονίζεται στην εικόνα 57.

Η επεξεργασία του άρθρου αποτελείται από μια σειρά βημάτων. Αρχικά πραγματοποιείται αναγνώριση οντοτήτων και στη συνέχεια ανάλυση συναισθήματος για κάθε οντότητα, με είσοδο στο κατάλληλο μοντέλο και τις προτάσεις που περιέχουν την οντότητα. Τέλος, κάθε οντότητα σχετίζεται με ένα άρθρο της ιστοσελίδας Wikipedia, μέσω μια αναζήτησης με χρήση τεχνικής που έχει αναφερθεί στην ενότητα 2.9 και τέλος αποθηκεύονται όλες αυτές οι πληροφορίες.

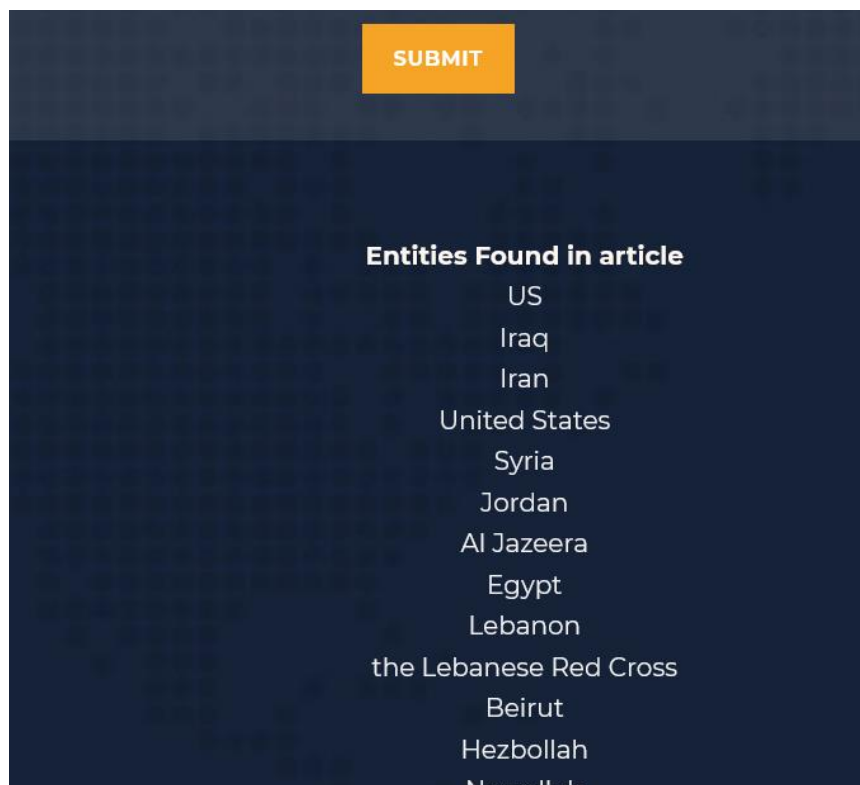


Σχήμα 55



Σχήμα 56

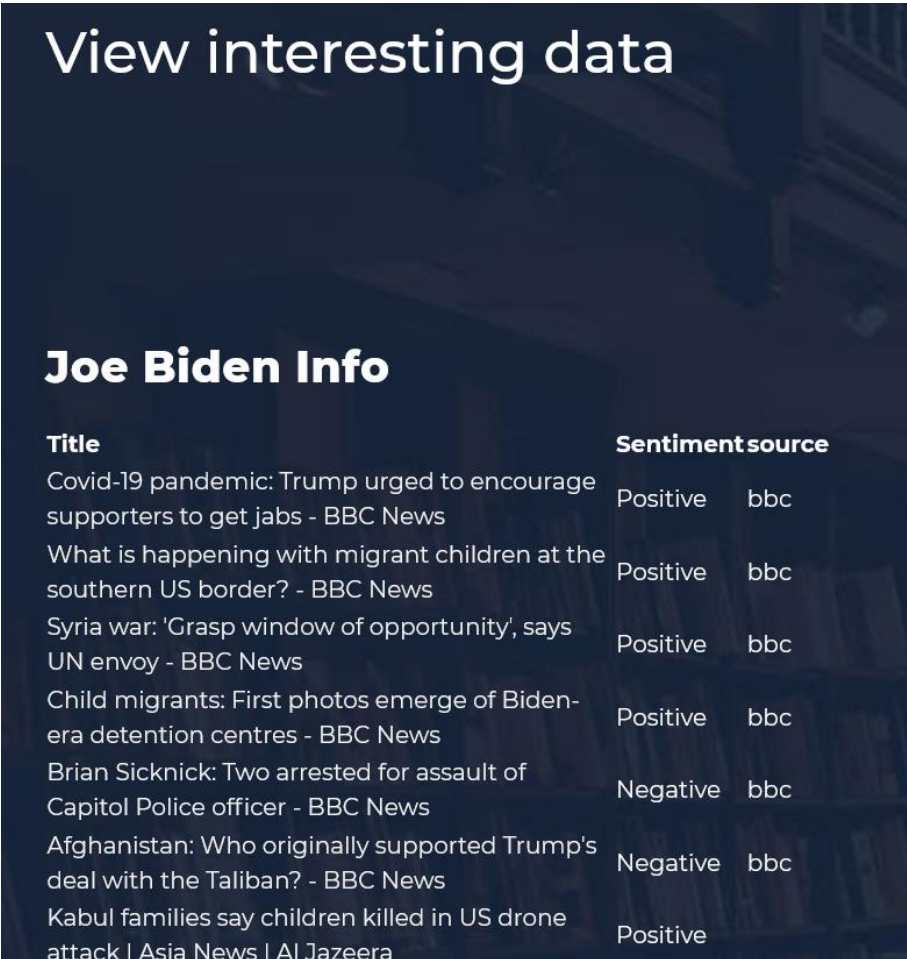
Αφού τελειώσει η επεξεργασία του άρθρου, εμφανίζονται όλες οι οντότητες που βρέθηκαν στο άρθρο. Οι οντότητες αποτελούν και σύνδεσμο που μεταφέρει το χρήστη σε σελίδα με περισσότερες πληροφορίες για την οντότητα.



Σχήμα 57

5. 8 Οντότητα

Στη σελίδα της οντότητας εμφανίζονται πληροφορίες μιας οντότητας. Συγκεκριμένα, εμφανίζονται σε μορφή πίνακα τα άρθρα που αναφέρουν την οντότητα, το συναίσθημα προς την οντότητα εντός του άρθρου και την πηγή του άρθρου. Ο τίτλος κάθε άρθρου είναι και σύνδεσμος που οδηγεί στη σελίδα του άρθρου. Ο χρήστης δεν οδηγείται στη σελίδα της οντότητας μόνο από την σελίδα υποβολής άρθρων, αλλά και από τη σελίδα απεικόνισης δεδομένων, κάτι που περιγράφεται στη παρακάτω ενότητα.



View interesting data		
Joe Biden Info		
Title	Sentiment source	
Covid-19 pandemic: Trump urged to encourage supporters to get jabs - BBC News	Positive	bbc
What is happening with migrant children at the southern US border? - BBC News	Positive	bbc
Syria war: 'Grasp window of opportunity', says UN envoy - BBC News	Positive	bbc
Child migrants: First photos emerge of Biden-era detention centres - BBC News	Positive	bbc
Brian Sicknick: Two arrested for assault of Capitol Police officer - BBC News	Negative	bbc
Afghanistan: Who originally supported Trump's deal with the Taliban? - BBC News	Negative	bbc
Kabul families say children killed in US drone attack Asia News Al Jazeera	Positive	

Σχήμα 58

5. 9 Δεδομένα

Στη σελίδα που φέρει την ονομασία MinedData, παρουσιάζονται ενδιαφέροντα δεδομένα προερχόμενα βέβαια από τα άρθρα που έχουν καταχωρηθεί στην εφαρμογή. Τα δεδομένα είναι διαχωρισμένα σε τέσσερις κατηγορίες, άνθρωποι, οργανισμοί, τοποθεσίες και πηγές.

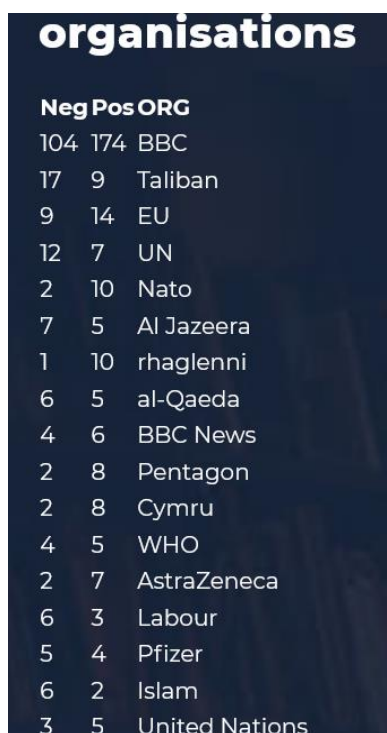


Σχήμα 59

Στο σχήμα 61, εμφανίζονται άνθρωποι ταξινομημένοι βάση του αριθμού αναφορών. Για κάθε άτομο εμφανίζεται η καταμέτρηση αρνητικών και θετικών αναφορών. Πατώντας στο όνομα του ατόμου, ο χρήστης μεταφέρεται στη σελίδα οντότητας που περιγράφεται στην ενότητα 5.8.



Σχήμα 60



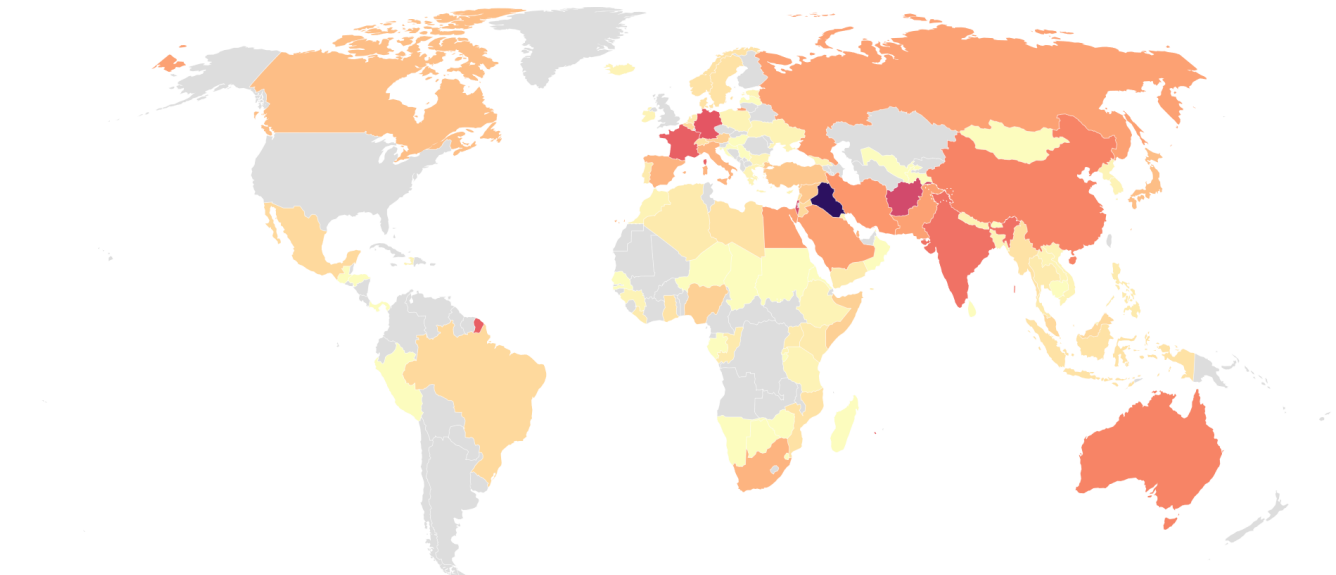
Σχήμα 61

Στο σχήμα 62 εμφανίζονται οι οργανισμοί με τις περισσότερες αναφορές και την καταμέτρηση θετικών αναφορών. Οι οργανισμοί, οι τοποθεσίες και οι άνθρωποι αποτελούν

οντότητες, όπως αναφέρεται και στην ενότητα 5.1. Με αυτόν τον τρόπο και πατώντας στην ονομασία, ο χρήστης μεταφέρεται στη σελίδα οντότητας.

Geographic locations & mentions

Stories
1 37



Source: News Archive • Created with Datawrapper

Σχήμα 62

Πατώντας την κατηγορία τοποθεσίες (Places), απεικονίζεται πίνακας με τις θετικές και αρνητικές αναφορές για κάθε τοποθεσία (Σχήμα 64). Εκτός από τον πίνακα εμφανίζεται και η εικόνα του σχήματος 63. Η εικόνα παρουσιάζει τον παγκόσμιο χάρτη, ο οποίος είναι χρωματισμένος σύμφωνα με το υπόμνημα, βάσει πλήθους αναφορών. Ακόμη, ο χάρτης έχει μια διαδραστική ιδιότητα, καθώς μεταφέροντας το ποντίκι “πάνω” από μια γεωγραφική περιοχή εμφανίζεται ο αριθμός αναφορών. Ο χάρτης δημιουργήθηκε με χρήση του προγράμματος Datawrapper.

Places		
Neg Pos Place		
54	50	US
41	35	UK
27	11	Iraq
19	13	Afghanistan
17	14	London
9	19	England
9	16	Washington
9	15	Germany
6	18	Australia
15	8	the United States

Σχήμα 63

Στην κατηγορία πηγές εμφανίζονται οι οντότητες ταξινομημένες βάσει πλήθους αναφορών και πηγής (Σχήμα 65). Ακόμα, εμφανίζεται η καταμέτρηση άρθρων ανά πηγή (Σχήμα 66).

Sources			
Neg	Pos	Entity	Source
101	165	BBC	bbc
5	9	EU	bbc
9	3	Taliban	bbc
6	5	Taliban	
1	10	rhaglenni	bbc
8	3	UN	aljazeera
2	8	Cymru	bbc
1	9	BBC	
2	7	Nato	bbc
3	6	BBC News	bbc
4	5	WHO	bbc
2	7	AstraZeneca	bbc
0	8	un	bbc
5	3	Labour	bbc
3	4	Pfizer	bbc
0	7	eu	bbc
6	1	the Metropolitan Police	bbc
5	2	Al Jazeera	
2	5	Oxford-AstraZeneca	bbc

Σχήμα 64

Articles Source	
1963	bbc
1872	aljazeera
21	

Σχήμα 65

5. 10 Πληροφορίες Εφαρμογής

Η σελίδα με τίτλο About us, περιέχει κείμενο που περιγράφει τη λειτουργία της εφαρμογής με απλούς όρους για τους ενδιαφερόμενους χρήστες.

Who we are

This application was created for people to view and archive articles and to be able explore interesting data extracted.

Once you have signed up you can create your own Libraries and store articles inside them. If you like an article and want to add it to our database you can do so by navigating to our submit page and simply pasting the url and hitting submit.

How we do it

When articles are submitted to the application they are passed through two processes.

The first process performs an entity recognition on the body of the article and then sentiment analysis for each entity. Entity Recognition is a technique that identifies people, organisations, places etc. inside articles. Sentiment analysis is the classification of a body of text regarding its sentiment or tone.

The second process finds related articles with the use of what is called a topic model, an algorithm created with the intention of finding underlying topics in a body of texts.

Σχήμα 66

5. 11 Επεξήγηση Κώδικα

```
def news_list(request):
    #print("news_list called")
    headlines=Property.objects.order_by('date')[:30]
    #print(headlines)
    headl=[]
    for item in headlines:
        if item.image_url!="www.noimage.com":
            headl.append(item)
    Topvideo=Property.objects.all().order_by('-views')[:1]
    print(Topvideo[0])
    context={
        'Topvideo':Topvideo,
        'object_list':headl,
    }
    return render(request,"news.html",context)
```

Σχήμα 67

Η παραπάνω συνάρτηση καλείται όταν ο χρήστης επιλέγει την αρχική σελίδα. Αρχικά, συλλέγονται τα τελευταία τριάντα άρθρα. Από τα άρθρα αυτά, αφαιρούνται εκείνα που δεν έχουν εικόνα. Ακόμη, αποθηκεύεται το άρθρο με τις περισσότερες προβολές. Τα δεδομένα αυτά στέλνονται στη συνάρτηση render που παράγει τη τελική σελίδα που βλέπει ο χρήστης.

```
def get_similar(request):
    lda=gensim.models.ldamodel.LdaModel.load("properties/lda/lda_15_articles_wlist/model15bodies")
    all_ids=Property.objects.values_list('id',flat=True).order_by('id')
    #print(all_ids)
    with open('properties/lda/lda_15_articles_wlist/list_ids15.pkl', 'rb') as f:
        ids = pickle.load(f)
    ids=list(ids)
    #print(ids)
    for j in all_ids:
        queryobject=Property.objects.filter(id=j).values()
        dic=queryobject[0]
        if not dic['similar_ids']:
            body=[]
            body.append(dic['body'])
            clean_body=clean.clean_text(body)
            vector1=lda[clean_body[0]]
            sims=clean.get_similarity(lda,vector1)
            #sims=clean.get_jensen_shannon(lda,vector1)
            sims = sorted(enumerate(sims), key=lambda item: -item[1])
            tens=sims[:10]
            articles=[]
            for i in range(10):
                index=tens[i][0]
                id=ids[index]
                articles.append(id)
            Property.objects.filter(id=j).update(similar_ids=articles)
    return render(request,"news.html")
```

Σχήμα 68

Στο σχήμα 69 προβάλλεται μια συνάρτηση που χρησιμοποιείται για την εύρεση και αποθήκευση των παρόμοιων άρθρων. Συγκεκριμένα, χρησιμοποιείται για άρθρα που έχουν εισαχθεί στη βάση αυτόματα, χωρίς να περάσουν από επεξεργασία. Εφόσον εισαχθούν πολλά άρθρα, για τα οποία δεν έχουν βρεθεί παρόμοια άρθρα,καλείται αυτή η συνάρτηση που βρίσκει και αποθηκεύει τα παρόμοια άρθρα, όπως περιγράφεται στην ενότητα 5.1.

```
def retrieve_article(request,id):-
    #scan_all_entities()
    update_viewcount(id)
    article_query=Property.objects.filter(id=id).values()
    context=article_query[0]
    similar_ids=context['similar_ids']
    similar=[]
    for sim_id in similar_ids:
        sim_article=Property.objects.values_list('name','image_url','id').filter(id=sim_id)
        similar.append(sim_article[0])
    context['similar']=similar
    if request.user.is_authenticated:
        mylibs=list(Library.objects.values_list('title',flat=True).filter(user=request.user).order_by('title'))
        context['library_names']=mylibs
    #print(context)
    return render(request,'article.html',context)

def update_viewcount(id):-
    article_views=Property.objects.values_list('views',flat=True).filter(id=id)
    article_views=article_views[0]+1
    Property.objects.filter(id=id).update(views=article_views)
```

Σχήμα 69

Η συνάρτηση `retrieve_article` καλείται με μεταβλητή το αναγνωριστικό του άρθρου και συλλέγει τα δεδομένα που προβάλλονται, όταν ο χρήστης επιλέγει ένα άρθρο. Δηλαδή, συλλέγει τα παρόμοια άρθρα και εφόσον ο χρήστης είναι συνδεδεμένος, συλλέγονται και οι βιβλιοθήκες του χρήστη.

```
def libraries(request):-
    context=None
    if request.user.is_authenticated:
        mylibs=list(Library.objects.values_list('id','title').filter(user=request.user).order_by('title'))
        context={'library_names':mylibs}
    return render(request,'libraries.html',context)
```

Σχήμα 70

Με την παραπάνω συνάρτηση ανακαλούνται οι βιβλιοθήκες του χρήστη και στη συνέχεια, προβάλλονται στη σελίδα με τις βιβλιοθήκες του χρήστη.

```
def create_library(request):
    if request.user.is_authenticated:
        if request.method=='POST':
            name=request.POST['name']
            print("NEW LIBRARY NAME ",name)
            print("user id ",request.user.id)

            if name != "":
                new_library=Library(title=name,user=request.user)
                new_library.save()
                mylibs=list(Library.objects.values_list('id','title').filter(user=request.user).order_by('title'))
                context={'library_names':mylibs}
            return render(request,'libraries.html',context)
```

Σχήμα 71

Η συνάρτηση στο σχήμα 72 καλείται όταν ο χρήστης δημιουργεί μια νέα βιβλιοθήκη. Αποθηκεύεται η νέα βιβλιοθήκη και έπειτα συλλέγονται οι βιβλιοθήκες και στέλνονται και πάλι για προβολή στη σελίδα.

```
def open_library(request,id):
    context={}
    if request.user.is_authenticated:
        opened_lib=Library.objects.get(user=request.user,id=id)
        articles=list(opened_lib.article_ids.all())
        context['article_names']=articles
        context['library']=opened_lib
    return render(request,'mylib.html',context)
```

Σχήμα 72

Η συνάρτηση `open_library`, με μεταβλητή το αναγνωριστικό της βιβλιοθήκης, συλλέγει τα άρθρα που έχουν καταχωρηθεί στη βιβλιοθήκη. Η συνάρτηση καλείται όταν ο χρήστης επιλέγει την προβολή μιας βιβλιοθήκης.

```
def add_to_lib(request,name,id):
    #get article id list and update it
    if request.user.is_authenticated:
        mylib=Library.objects.get(user=request.user,title=name)
        Article=Property.objects.get(id=id)
        mylib.article_ids.add(Article)
        mylib.save()
        print(mylib)
    return retrieve_article(request,id)
```

Σχήμα 73

```
def delete_library(request,id):
    if request.user.is_authenticated:
        mylib=Library.objects.filter(user=request.user,id=id).delete()
    return libraries(request)
```

Σχήμα 74

Η `add_to_lib` έχει δύο μεταβλητές: το όνομα της βιβλιοθήκης και το αναγνωριστικό του άρθρου. Λειτουργία της συνάρτησης είναι η προσθήκη ενός άρθρου σε μία βιβλιοθήκη.

Η `delete_library` διαγράφει μια βιβλιοθήκη και στη συνέχεια, επιστρέφει στη σελίδα των βιβλιοθηκών του χρήστη.

```

def find_similar(bod):
    lda=gensim.models.ldamodel.LdaModel.load('properties/lda/lda_15_articles_wlist/model15bodies')
    all_ids=Property.objects.values_list('id',flat=True).order_by('id')
    with open('properties/lda/lda_15_articles_wlist/list_ids15.pkl', 'rb') as f:
        ids = pickle.load(f)
    ids=list(ids)
    body=[]
    body.append(bod)
    clean_body=clean.clean_text(body)
    vector1=lda[clean_body[0]]
    sims=clean.get_similarity(lda,vector1)
    #sims=clean.get_jensen_shannon(lda,vector1)
    sims = sorted(enumerate(sims), key=lambda item: -item[1])
    tens=sims[:10]
    articles=[]
    for i in range(10):
        index=tens[i][0]
        id=ids[index]
        articles.append(id)
    #Property.objects.filter(id=j).update(similar_ids=articles)
    return articles

```

Σχήμα 75

Η λειτουργία της παραπάνω συνάρτησης είναι η εύρεση παρόμοιων άρθρων με το εισαγόμενο κείμενο. Δέχεται ως είσοδο το κείμενο ενός άρθρου.Εν συνεχεία καλείται η συνάρτηση `clean_text`, η οποία προεπεξεργάζεται το κείμενο, ώστε να δοθεί ως είσοδος στο μοντέλο `lda` στη συνέχεια. Τέλος, χρησιμοποιείται η συνάρτηση `get_similarity`, η οποία βρίσκει τα πιο παρόμοια άρθρα και επιστρέφεται η λίστα με τα αναγνωριστικών τους. Η συνάρτηση καλείται κατά την υποβολή του άρθρου στην εφαρμογή.

```

def clean_text(text):
    data = text
    data = [re.sub('\S*\s*\s?', '', sent) for sent in data]
    data = [re.sub('\s+', ' ', sent) for sent in data]
    data = [re.sub("\'", "", sent) for sent in data]
    #print('DATA AFTER FIRST PART ',data)
    data_words = list(sent_to_words(data))
    def remove_stopwords(texts):
        stop_words = stopwords.words('english')
        return [[word for word in simple_preprocess(str(doc)) if word not in stop_words] for doc in texts]
    def lemmatization(texts, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):
        nlp = spacy.load('en_core_web_sm', disable=['parser', 'ner'])
        texts_out = []
        for sent in texts:
            doc = nlp(" ".join(sent))
            texts_out.append([token.lemma_ for token in doc if token.pos_ in allowed_postags])
        return texts_out
    data_words_nostops = remove_stopwords(data_words)
    data_lemmatized = lemmatization(data_words_nostops, allowed_postags=[
        'NOUN', 'ADJ', 'VERB', 'ADV'
    ])
    #print('data_lemmatized',data_lemmatized)
    dict = corpora.Dictionary.load('properties/lda/lda_15_articles_wlist/model15bodies.id2word')
    converted = [dict.doc2bow(text) for text in data_lemmatized]
    #print('data ready ',converted)
    return converted

```

Σχήμα 76

Η συνάρτηση στο σχήμα 77, επεξεργάζεται κείμενο με σκοπό να χρησιμοποιηθεί σε μοντέλο `lda`. Αρχικά αφαιρούνται συνδετικές λέξεις και στη συνέχεια οι καταλήξεις.Μετά, το κείμενο αναπαρίσταται λέξη προς λέξη, με το λεξικό του μοντέλου (το μοντέλο έχει εκπαιδευτεί με ένα μεγάλο σώμα κειμένων, των οποίων τις λέξεις έχει σε ένα λεξικό και έχει αντιστοιχίσει κάθε λέξη με έναν αριθμό). Η έξοδος αυτής της διαδικασίας είναι έτοιμη για να δοθεί στο μοντέλο `lda`.

```
def get_similarity(lda, query_vector):
    print("reached")
    corpus=0
    with open('properties/lda/lda_15_articles_wlist/corpus15bodies', 'rb') as f:
        corpus = pickle.load(f)
    index = similarities.MatrixSimilarity(lda[corpus])
    sims = index[query_vector]
    return sims
```

Σχήμα 77

Στην συνάρτηση `get_similarity` δίνεται ως είσοδος ένα μοντέλο `lda` και μια έξοδος του μοντέλου (`vector1` είναι η έξοδος του μοντέλου `lda` με είσοδο ενός κειμένου). Χρησιμοποιώντας την `MatrixSimilarity`, με είσοδο το διάνυσμα, επιστρέφεται η λίστα διανυσμάτων με την ομοιότητα.

```
def view_entity_data(request,ent_id):
    cursor=connection.cursor()
    entity_data="select t1.id,t1.name,t2.sentiment,t1.source from properties_property as t1,properties_sentiment as t2,properties_entity as t3 where t2.article_id=t1.id and t2.entity_id=id and t3.id=id)".format(id=ent_id)
    cursor.execute(entity_data, ['localhost'])
    entity_data= cursor.fetchall()
    ent="select name from properties_entity where id={id}".format(id=ent_id)
    cursor.execute(ent, ['localhost'])
    entity=cursor.fetchall()
    context={"entity_data":entity_data,"entity_name":entity}
    print(entity)
    return render(request,'entity.html',context)
```

Σχήμα 78

Στη συνάρτηση `view_entity_data` συλλέγονται τα δεδομένα μιας οντότητας μέσω ερωτημάτων SQL που δεν μπορούν να δημιουργηθούν μέσω του συστήματος ORM.

```
def named_entity_recognition(id):
    text_body =Property.objects.filter(id=id).values_list('body',flat=True)
    Article_instance = Property.objects.get(id=id)
    entities,types,urls=process_for_ner(text_body[0])
    for ent in entities.keys():
        new_entity=Entity.objects.get_or_create(name=ent,type=types[ent])
        sentiment=get_sentiment(entities[ent])
        Sentiment.objects.get_or_create(article=Article_instance,entity=new_entity[0],sentiment=sentiment)
        for url in urls[ent]:
            if url is not None:
                wiki=Wikipedia_url.objects.get_or_create(url=url)
                wiki[0].entity.add(new_entity[0])
    except:
        pass
```

Σχήμα 79

Στο σχήμα 54 η συνάρτηση `named_entity_recognition` εκτελεί τη διαδικασία αναγνώρισης οντοτήτων. Με τη συνάρτηση `process_for_ner` βρίσκονται οι οντότητες και οι σύνδεσμοι τους στη wikipedia και στη συνέχεια κάθε οντότητα αποθηκεύεται στη βάση, ελέγχοντας αν υπάρχει ήδη ώστε να μην υπάρξουν διπλές εισαγωγές.

```
def process_for_ner(doc):
    doc=nlp(doc)
    entities=[]
    for ent in doc.ents:
        if ent.label_ in ("GEO","GPE","ORG","PERSON"):#
            entities.append((ent.text,ent.label_))#
    #Remove Duplicates#
    entities=list(set(entities))#
    #print(entities)#
    #Create a dictionary with entities as keys and a list of sentences as values#
    ent_sentences={}#
    #for each entity we calculate the Levenstein ratio with the other entities#
    all_candidates=[]
    entity_types={}
    entity_urls={}
    for ent in entities:#
        ratio=[]#
        #print(ent[0])#
        for i in range(len(entities)):#
            #check only entities with the same type#
            if ent[1] == entities[i][1]:#
                ratio.append((levenstein(ent[0],entities[i][0]),entities[i][0]))#
            #keep entities with score higher than 80(assuming the differences between 80-100 are meaningless)#
            #also sorting the resulting list to later get the first entity as the final one#
            candidates=sorted([element[1] for element in ratio if element[0]>80 ],reverse=True)#
            print(candidates)#
            #add candidates in a list of lists to keep each list only once#
            if candidates not in all_candidates:#
                all_candidates.append(candidates)#
                entity_types[candidates[0]]=ent[1]#
            for candidates in all_candidates:#
                #here we gather all wikipedia links related to this page#
                urls=[get_wikipedia_url(candidate) for candidate in candidates]#
                entity_urls[candidates[0]]=urls#
            #make list of lists of sentences for each candidate#
            sentences=[find_sents(doc,candidate) for candidate in candidates]#
            #turn it into regular list and remove duplicates#
            all_sentences=list(set([item for sublist in sentences for item in sublist]))#
            #Create a dictionary associating each final key with the sentences to be later used to get the sentiment score#
            ent_sentences[candidates[0]]=all_sentences#
    return ent_sentences,entity_types,entity_urls#
```

Σχήμα 80

Στη συνάρτηση `process_for_ner` δίνεται το κείμενο ενός άρθρου ως είσοδος. Για κάθε οντότητα που αναγνωρίζεται εντός του κειμένου, χρησιμοποιείται η συνάρτηση `levensthein` που παράγει μια τιμή ομοιότητας για κάθε άλλη οντότητα. Οντότητες με πολύ υψηλή ομοιότητα, θεωρούνται ως η ίδια οντότητα. Στη συνέχεια χρησιμοποιείται η `get_wikipedia_url` ώστε να βρεθούν σελίδες της wikipedia για κάθε οντότητα. Τέλος επιστρέφονται οι οντότητες, ο τύπος τους και οι σύνδεσμοι.

[illegible]

Σχήμα 81

Η συνάρτηση topics επιστρέφει τα δεδομένα που προβάλλονται στη σελίδα MinedData, με μια σειρά ερωτημάτων SQL.

```

def submit(request):
    print(request.POST)
    if request.method=='POST' and request.POST.get('title') is None:
        print("entered URL SUBMIT")
        url=request.POST['url']
        session=requests.Session()
        session.headers={"User-Agent": "Googlebot/2.1 (http://www.google.com/bot.html)"}
        content=session.get(url,verify=True).content
        soup=BSoup(content,"html.parser")
        dom = etree.HTML(str(soup))
        #print(dom.xpath('//title/text()')[0])
        #print(dom.xpath('//p/text()')[0])
        img_url=dom.xpath("//img/@src")[0]
        source = re.sub("(.*)", "", url)
        source = re.sub("(.*)", "", source)
        context={'name':dom.xpath('//title/text()')[0], 'body': ' '.join(dom.xpath('//p/text()')), 'url':url, 'image_url':dom.xpath("//img/@src")[0], 'source':source}
        return render(request, 'subm.html', context)
    #new_article=Property(name=dom.xpath('//title/text()')[0], body=' '.join(dom.xpath('//p/text()')), url=url, image_url=img_url)
    #new_article.save()
    #named_entity_recognition(new_article.id)
    #return render(request, 'subm.html')
    elif request.POST.get('title'):
        new_article=Property(name=request.POST['title'], body=request.POST['body'], url=request.POST['url'], image_url=request.POST['image_url'])
        print("SAVING ARTICLEEEEEEEEE")
        new_article.save()
        similar=find_similar(request.POST['body'])
        Property.objects.filter(id=new_article.id).update(similar_ids=similar)
        named_entity_recognition(new_article.id)
        #retrieve computed entities
        cursor=connection.cursor()
        query="select t1.entity_id,t2.name from properties_sentiment as t1,properties_entity as t2 where article_id={} and t1.entity_id=t2.id".format(new_article.id)
        cursor.execute(query,['localhost'])
        entities=cursor.fetchall()
        #entities=[item[1] for item in entities]
        print(entities)
        context={'entities':entities}
        return render(request, 'subm.html', context)
    else:
        print('ERROR WITH FORM')
        return render(request, 'subm.html')

```

Σχήμα 82

Η συνάρτηση submit χρησιμοποιείται για την υποβολή των άρθρων. Αποτελείται από δυο μέρη. Το πρώτο μέρος εκτελείται όταν ο χρήστης καταχωρεί τον σύνδεσμο του άρθρου. Με χρήση της βιβλιοθήκης BeautifulSoup συλλέγονται τα δεδομένα του άρθρου και στέλνονται πίσω στο χρήστη για επιβεβαίωση. Το δεύτερο μέρος της συνάρτησης χρησιμοποιείται όταν ο χρήστης επιβεβαιώσει τις πληροφορίες που συλλέχθηκαν. Τότε αποθηκεύεται το άρθρο και ξεκινά η ανάλυση του άρθρου με χρήση της process_for_ner. Όταν ολοκληρωθεί η ανάλυση του άρθρου επιστρέφονται στο χρήστη οι οντότητες με συνδέσμους προς τα δεδομένα τους.

```

def get_sentiment(sentences):
    sequences=process_for_sentiment(sentences)
    #print(sequences)
    predictions=model.predict(sequences)
    #print(predictions)
    classes = (predictions>0.25)
    #print(classes)
    if sum(classes==True)>len(classes)/2:
        return True
    else:
        return False

```

Σχήμα 83

Με την get_sentiment κατηγοριοποιούνται οι προτάσεις σε θετικές ή αρνητικές.

Αρχικά γίνεται επεξεργασία των προτάσεων και στην συνέχεια υπολογίζεται η κατηγορία που ανήκουν.

5.12 Εγκατάσταση Εφαρμογής

Για την εγκατάσταση της εφαρμογής συνίσταται η χρήση ενός εικονικού περιβάλλοντος, για την εγκατάσταση των βιβλιοθηκών. Εντός του περιβάλλοντος, αρχικά θα πρέπει να εγκατασταθούν οι βιβλιοθήκες του κεφαλαίου 3. Αφότου εγκατασταθούν, θα πρέπει ο χρήστης να κατεβάσει το αποθετήριο από το github (Repository) και τη βάση δεδομένων PostgreSQL, όπως και τη βάση από το dropbox (Database file). Ακόμη, θα πρέπει να κατεβάσει το πρόγραμμα pgAdmin, που χρησιμεύει στη διαχείριση της βάσης. Με το pgAdmin θα πρέπει ο χρήστης να κάνει εισαγωγή της βάσης στο postgresSQL. Τέλος, για την ενεργοποίηση της εφαρμογής θα πρέπει ο χρήστης να ανοίξει ένα τερματικό, να μεταβεί στο φάκελο main του αποθετηρίου και να εκτελέσει την εντολή **python manage.py runserver**.

Κεφάλαιο 6 Συμπεράσματα και Μελλοντικές επεκτάσεις

Η ανάπτυξη της εφαρμογής δείχνει τις δυνατότητες της μηχανικής μάθησης και τους τρόπους, που μπορεί να χρησιμοποιηθεί για να επεκτείνει τις γνώσεις, αλλά και την ικανότητα του χρήστη, να χρησιμοποιεί το διαδίκτυο ως μέσο ενημέρωσης. Συλλέγοντας πληροφορίες για τις ίδιες της πηγές, ο χρήστης μπορεί πιο εύκολα να διακρίνει διαφορές στην αφήγηση των ειδήσεων, μεταξύ πηγών. Η βελτίωση της ενημέρωσης του αναγνώστη με εργαλεία, όπως αυτό που περιεγράφηκε παραπάνω, μπορεί να συγκριθεί με κάποιες άλλες πρακτικές που ακολουθούνται πλέον, από διάφορα μέσα κοινωνικής δικτύωσης. Στα μέσα αυτά χρησιμοποιούνται επίσης, μοντέλα μηχανικής μάθησης με σκοπό όμως την εύρεση ψευδών ειδήσεων ή μη επιτρεπτού λεκτικού περιεχομένου, γενικότερα. Οι δύο αυτές χρήσεις της μηχανικής μάθησης, αποτελούν διαφορετικές μεθόδους εκτέλεσης του ίδιου σκοπού, δηλαδή την καταπολέμηση της παραπληροφόρησης. Αυτές οι μέθοδοι δεν έχουν μόνο πρακτικές διαφορές αλλά και ηθικές, καθώς στη πρώτη διευκολύνεται το έργο του αναγνώστη, ενώ στη δεύτερη υπάρχει περιορισμός του σε πρόσβαση κάποιων πληροφοριών, που έχουν θεωρηθεί ψευδείς από ένα τρίτο πρόσωπο, αναπαράγοντας έτσι τη δική του έκδοση της πραγματικότητας.

Υπάρχουν πολλά ακόμη που θα μπορούσαν να προστεθούν στην εφαρμογή, που θα επέκτειναν την χρησιμότητα της για τον χρήστη. Μια χρήσιμη λειτουργία θα ήταν ο διαμοιρασμός προσωπικών βιβλιοθηκών άρθρων προς τρίτους, που δεν είναι χρήστες της εφαρμογής. Ο διαμοιρασμός θα μπορούσε να γίνει ακόμη με τη αυτόματη δημιουργία ενός αρχείου, που να περιλάμβανε όλα τα άρθρα. Για τη διευκόλυνση του χρήστη, θα μπορούσε να δημιουργηθεί πρόσθετο (plugin) για τα προγράμματα περιήγησης, το οποίο να επιτρέπει την αυτόματη υποβολή ενός άρθρου στην εφαρμογή. Τέλος, μια χρήσιμη λειτουργία θα ήταν η εμφάνιση προτεινόμενων άρθρων.

Bibliography

- Bo Pang-Lillian Lee-Peter Turney. (2002). Thumbs up? Sentiment Classification using Machine Learning. Retrieved 04 2021, from <https://www.cs.cornell.edu/home/llee/papers/sentiment.pdf>
- Database file. (n.d.). Retrieved from https://www.dropbox.com/s/2d8nqznuq5s1vub/db_thesis111.7z?dl=0
- Gerald Stalton. (1972). A Statistical Interpretation of term specificity and its application in retrieval. Retrieved 08 2021, from <https://www.emerald.com/insight/content/doi/10.1108/eb026526/full/html>
- Ho, T. K. (1995). Random Decision Forests. Retrieved 04 2020, from <https://web.archive.org/web/20160417030218/http://ect.bell-labs.com/who/tkh/publications/papers/odt.pdf>
- Jonathan K. Pritchard-Matthew Stephens-Peter Donnelly. (2000). Inference of Population Structure Using Multilocus Genotype Data. Retrieved 3 2020, from <https://www.genetics.org/content/155/2/945>
- Karen Spärck Jones. (1972). https://web.archive.org/web/20110116062252/http://www.soi.city.ac.uk/~ser/idfpapers/ksj_orig.pdf. Retrieved 09 2020, from https://web.archive.org/web/20110116062252/http://www.soi.city.ac.uk/~ser/idfpapers/ksj_orig.pdf
- M. Stone . (1974). Cross-Validatory Choice and Assessment of Statistical Predictions . Retrieved from <https://www.jstor.org/stable/2984809>
- McCallum, D. M.-E.-M.-H.-A. (2011). Optimizing Semantic Coherence in Topic Models. Retrieved from <https://mimno.infosci.cornell.edu/papers/mimno-semantic-emnlp.pdf>
- Repository. (n.d.). Retrieved from https://github.com/fabianhoegger/Article_App
- Stanford Sentiment Treebank. (n.d.). Retrieved from <https://www.kaggle.com/atulanandjha/stanford-sentiment-treebank-v2-sst2>
- Twitter US Airline Sentiment. (n.d.). Retrieved from <https://www.kaggle.com/crowdflower/twitter-airline-sentiment>

