Προγραμματιστική Εργασία: Επέκταση Ερωτημάτων με Συνώνυμους Όρους για τη Βελτίωση των Αποτελεσμάτων της Ανάκτησης

Ο σκοπός της εργασίας είναι να εξασκηθείτε σε κλασικές μεθόδους και μοντέλα ανάκτησης πληροφορίας, αλλά και να εφαρμόσετε state-of-the-art τεχνικές για να βελτιώσετε τα αποτελέσματα ενός συστήματος ανάκτησης πάνω σε πραγματικά δεδομένα.

Ένα από τα πιο σημαντικά βήματα δημιουργίας ενός συστήματος ανάκτησης πληροφορίας είναι η εφαρμογή αλγορίθμων ανάλυσης κειμένου στη συλλογή δεδομένων. Οι αλγόριθμοι αυτοί προσδιορίζουν τον τρόπο με τον οποίο γίνεται η επεξεργασία του κειμένου μας και προκύπτουν οι όροι που θα μπουν στο ευρετήριό μας. Το ερώτημα του χρήστη αφού υποβληθεί θα υποστεί αντίστοιχη επεξεργασία και οι όροι του θα συγκριθούν με τους όρους του ευρετηρίου. Τα κείμενα που περιέχουν τους όρους του ερωτήματος θα επιστραφούν ως συναφή στο χρήστη.

Ένα από τα μεγαλύτερα προβλήματα στη διαδικασία αυτή είναι ότι οι χρήστες μπορεί να εκφράσουν την πληροφοριακή τους ανάγκη με διαφορετικούς τρόπους. Για παράδειγμα, το "walk in the mountains" μπορεί να εκφραστεί και ως "trekking" ή "hiking". Αν ο χρήστης υποβάλει το ερώτημα "hiking", αλλά το ευρετήριο περιέχει κείμενα με τη λέξη trekking, τα κείμενα αυτά δεν θα επιστραφούν στο χρήστη, παρόλο που είναι συναφή. Ο χρήστης είναι πιθανό να μην ικανοποιήσει την πληροφοριακή του ανάγκη. Ένας τρόπος για να αντιμετωπιστεί το πρόβλημα αυτό είναι το σύστημα ανάκτησης να γνωρίζει τα συνώνυμα των όρων του ερωτήματος.

Καλείστε να δημιουργήσετε μια μηχανή αναζήτησης, η οποία θα χρησιμοποιεί την τεχνική της επέκτασης-διεύρυνσης του ερωτήματος του χρήστη με συνώνυμους όρους έτσι ώστε να μπορεί να εκφράσει την πληροφοριακή ανάγκη του χρήστη με διαφορετικούς τρόπους και να αντιμετωπίσει προβλήματα όπως αυτό που περιγράφηκε παραπάνω.

Θα εφαρμόσετε δύο τρόπους για να βρίσκετε συνώνυμα. Ο ένας είναι με χρήση ειδικού λεξικού συνώνυμων όρων και ο άλλος με χρήση του δημοφιλή αλγόριθμου νευρωνικών δικτύων word2vec. Κάθε μεθοδολογία έχει τα δικά της πλεονεκτήματα, αλλά και τους δικούς της περιορισμούς.

Τέλος, θα αξιολογήσετε τη μηχανή αναζήτησής σας πάνω στη συλλογή `IR2020` χρησιμοποιώντας το εργαλείο αξιολόγησης `trec_eval`.

Φάση 1 – Baseline

Η βάση `IR2020` είναι μία συλλογή από 18.316 κείμενα. Περιλαμβάνει ένα σύνολο από ερωτήματα μαζί με τις σωστές συναφείς απαντήσεις.

1. Προεπεξεργαστείτε τη συλλογή (αρχείο `documents.txt`) προκειμένου να είναι σε κατάλληλη μορφή για να χρησιμοποιηθεί από τη μηχανή αναζήτησης `Lucene`.
2. Δημιουργήστε ένα ευρετήριο από τη συλλογή χρησιμοποιώντας τη μηχανή αναζήτησης `Lucene`. Επιλέξτε κατάλληλο `Analyzer` και συνάρτηση ομοιότητας.

3. Εκτελέστε τα ερωτήματα (αρχείο `queries.txt`) πάνω στο ευρετήριο και συλλέξτε τις απαντήσεις της μηχανής, τα k πρώτα ανακτηθέντα κείμενα, για k=20, 30, 50.
4. Αξιολογήστε τις απαντήσεις σας συγκρίνοντάς τες με τις σωστές απαντήσεις (αρχείο `qrels.txt`) χρησιμοποιώντας το εργαλείο αξιολόγησης `trec_eval` και τα μέτρα αξιολόγησης MAP (mean average precision) και avgPre@k (μέση ακρίβεια στα k πρώτα ανακτηθέντα κείμενα) για k=5, 10, 15, 20.
5. Καταγράψτε τα πειράματά σας σε μια αναφορά. Περιγράψτε πώς υλοποιήσατε τα 4 παραπάνω βήματα, συμπεριλάβετε screenshots όπου θεωρείτε χρήσιμο, και φτιάξτε έναν πίνακα με τα αποτελέσματα του `trec_eval` για τις διάφορες τιμές του k. Δημιουργήστε ένα αρχείο pdf με την αναφορά σας. Θα υποβάλλετε την αναφορά σας, τον κώδικά σας και τα αποτελέσματα του `trec_eval` σε ένα αρχείο `zip` με ονομασία `αριθμός_μητρώου.zip` (πχ. 3950000.zip). Μη συμπεριλάβετε την `IR2020`. Καταγράψτε τις πηγές σας.

Φάση 2 – Επέκταση ερωτήματος με συνώνυμα από το `WordNet`

Το `WordNet` είναι μια λεξική βάση δεδομένων για την αγγλική γλώσσα. Ομαδοποιεί τις αγγλικές λέξεις σε σύνολα συνωνύμων που ονομάζονται *synsets*, παρέχει σύντομους ορισμούς και παραδείγματα χρήσης των λέξεων και καταγράφει ορισμένες σχέσεις μεταξύ αυτών των συνόλων ή των μελών τους.

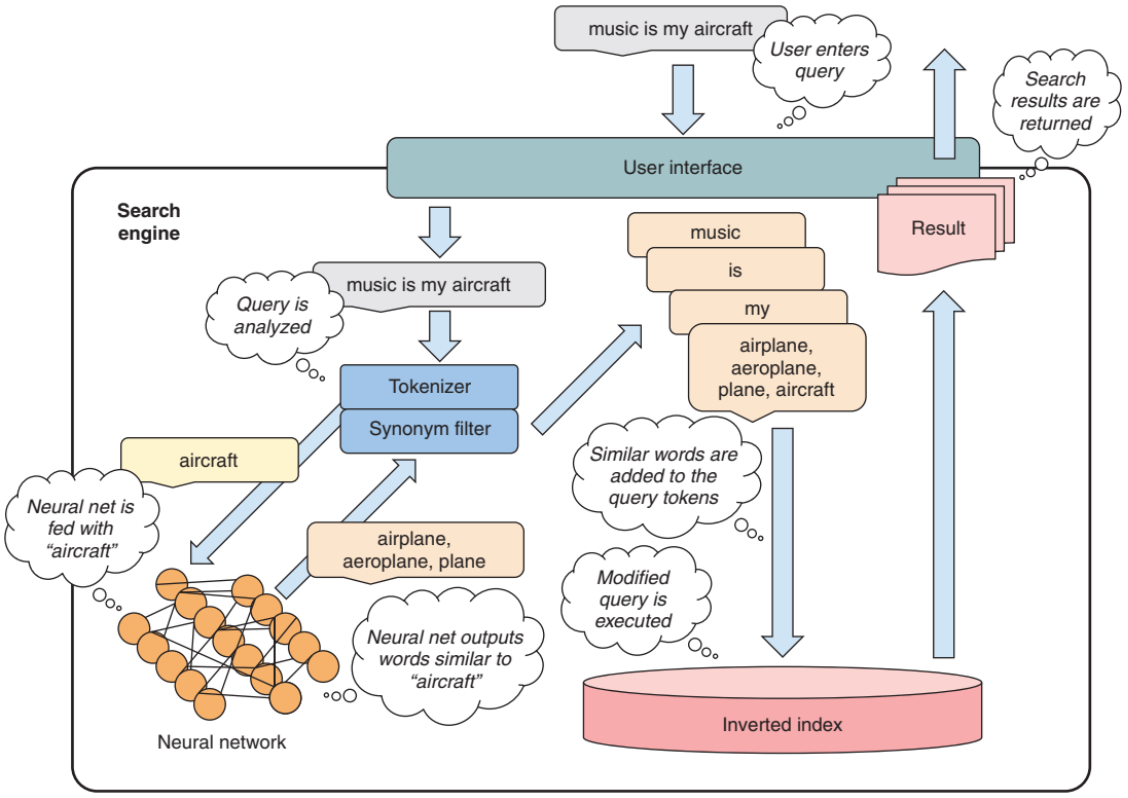
1. Κατεβάστε τα συνώνυμα του `WordNet` (αρχείο `wn_s.pl` στο eclass).
2. Επεκτείνετε τα ερωτήματα της `IR2020` με τους συνώνυμους όρους από το `WordNet`.
    a. Διαβάστε τα συνώνυμα και αποθηκεύστε τα σε μια δομή `Map`.
    b. Δημιουργήστε έναν `CustomAnalyzer` κατά την αναζήτηση, ο οποίος θα αναλύει τα ερωτήματα της συλλογής `IR2020` σε tokens και ελέγχοντας τη δομή του `Map` θα επεκτείνει κάποια tokens με το ή τα συνώνυμά τους εφόσον υπάρχουν (δείτε κώδικα 1.0 για παράδειγμα). Μπορείτε να χρησιμοποιήσετε και το `package` που παρέχει η `Lucene` για τη λήψη συνώνυμων από το `WordNet` (δείτε τις οδηγίες εδώ): `org.apache.lucene.analysis.synonym.WordnetSynonymParser`
3. Επαναλάβετε τα βήματα 3 έως 5 της Φάσης 1 για τα επαυξημένα με συνώνυμα ερωτήματά σας. Στην αναφορά σας συγκρίνετε τα αποτελέσματα της Φάσης 2 με της Φάσης 1. Υπάρχει κάποια βελτίωση; Προσπαθήσετε να αιτιολογήσετε τα αποτελέσματά σας είτε αυτά είναι θετικά είτε αρνητικά.

```
Map<String, String> sffargs = new HashMap<>();

sffargs.put("synonyms", "synonyms-wn.txt");

sffargs.put("format", "wordnet");

CustomAnalyzer.Builder builder = CustomAnalyzer.builder()

       .withTokenizer(WhitespaceTokenizerFactory.class)

       .addTokenFilter(SynonymGraphFilterFactory.class, sffargs)

return builder.build();
```

Κώδικας 1.0: Χρησιμοποιώντας τα συνώνυμα του `WordNet`

Φάση 3 – Επέκταση ερωτήματος με συνώνυμα από το `word2vec`

Το `word2vec` είναι ένας αλγόριθμος που βασίζεται σε νευρωνικά δίκτυα πρόσθιας τροφοδότησης για τη μάθηση διανυσματικών αναπαραστάσεων των λέξεων που μπορούν να χρησιμοποιηθούν για την εύρεση λέξεων με παρόμοια σημασία ή λέξεων που εμφανίζονται σε παρόμοια περιβάλλοντα (contexts). Το μοντέλο εκτιμά την πιθανότητα να επιλεχθεί μία λέξη (output) με βάση το περιβάλλον της (input). Εξάγει τους κοντινότερους γείτονες μιας λέξης εξετάζοντας τα συμφραζόμενα, το περιβάλλον της λέξης και καθορίζει πότε δύο λέξεις είναι σημασιολογικά συναφείς (όταν εμφανίζονται σε ίδιο ή παρόμοιο περιβάλλον-context). Υπό το πρίσμα αυτό μπορούμε να χρησιμοποιήσουμε το μοντέλο για να ανακαλύψουμε συνώνυμες λέξεις.



1. Εκπαιδεύστε ένα μοντέλο `word2vec` χρησιμοποιώντας τη συλλογή `IR2020` ως είσοδο και τη βιβλιοθήκη `DeepLearningForJava` (`DL4J`), η οποία παρέχει υλοποιημένα μοντέλα νευρωνικών δικτύων, όπως το `word2vec`, για τη `java`. Η διαδικασία της παραγωγής του μοντέλου μπορεί να παραμετροποιηθεί ως προς την αρχιτεκτονική που θα χρησιμοποιηθεί, το μέγεθος παραθύρου ελέγχου και τον αριθμό των διαστάσεων. Οι διαθέσιμες αρχιτεκτονικές είναι οι `Skip-gram` και `CBOW` που θα αναλυθούν στο μάθημα. Το παράθυρο ελέγχου ορίζει τον αριθμό των λέξεων που θα λαμβάνονται υπόψη, πριν και μετά από την κάθε λέξη. Όσον αφορά τον αριθμό των διαστάσεων, αυτός ορίζει την πολυπλοκότητα και το μέγεθος του μοντέλου.
2. Επεκτείνετε τα ερωτήματα της `IR2020` με τους συνώνυμους όρους από το μοντέλο που κατασκευάσατε. Δημιουργήστε έναν `CustomAnalyzer` κατά την αναζήτηση, ο οποίος θα αναλύει τα ερωτήματα της συλλογής `IR2020` σε tokens και ελέγχοντας το εκπαιδευμένο

μοντέλο `word2vec` θα επεκτείνει κάποια tokens με το ή τα συνώνυμά τους εφόσον υπάρχουν (δείτε κώδικα 2.0 για παράδειγμα ή δείτε εδώ).

3. Επαναλάβετε τα βήματα 3 έως 5 της Φάσης 1 για τα επαυξημένα με συνώνυμα ερωτήματά σας. Στην αναφορά σας συγκρίνετε τα αποτελέσματα της Φάσης 3 με της Φάσης 1 και 2. Υπάρχει κάποια βελτίωση; Προσπαθήσετε να αιτιολογήσετε τα αποτελέσματά σας είτε αυτά είναι θετικά είτε αρνητικά.

```
String filePath = new ClassPathResource(
       "my_corpus.txt").getFile()
       .getAbsolutePath();
SentenceIterator iter = new BasicLineIterator(filePath);


Word2Vec vec = new Word2Vec.Builder()
       .layerSize(100)
       .windowSize(5)
       .iterate(iter)
       .elementsLearningAlgorithm(new CBOW<>())
       .build();
vec.fit();

String[] words = new String[]{"here", "go", "the", "terms", "of", "the",
"query"};
for (String w : words) {
       Collection<String> lst = vec.wordsNearest(w, 2);
       System.out.println("2 Words closest to '" + w + "': " + lst);
}
```

Κώδικας 2.0: DL4J `word2vec` παράδειγμα

ΠΡΟΣΟΧΗ ΣΤΙΣ ΦΑΣΕΙΣ 2 ΚΑΙ 3
Η επέκταση των ερωτημάτων με συνώνυμους όρους πρέπει να γίνει με *προσοχή* τόσο στη Φάση 2 όσο και στη Φάση 3. Θα πρέπει να "ελέγξετε" τη διαδικασία διεύρυνσης έτσι ώστε να αποδίδονται συνώνυμα μόνο σε ορισμένους από τους όρους του ερωτήματος. Δεν έχουν όλοι οι όροι ενός ερωτήματος την ίδια σημασία. Για παράδειγμα, δεν ενδείκνυται να αποδώσετε συνώνυμους όρους στις τετριμμένες λέξεις. Τα αποτελέσματα μπορεί να χειροτερέψουν. Στη Φάση 2 επιλέξτε τα συνώνυμα του `WordNet` για ορισμένα μόνο μέρη του λόγου. Για παράδειγμα, μπορείτε να χρησιμοποιήσετε συνώνυμα μόνο για τα ουσιαστικά ή τα επίθετα και όχι για τα ρήματα. Στη Φάση 3 επιλέξτε τα συνώνυμα με τη μεγαλύτερη ομοιότητα ή με ομοιότητα μεγαλύτερη από ένα threshold. Επίσης, μπορείτε να ορίσετε συνώνυμα μόνο για λέξεις του ερωτήματος που έχουν μεγάλο βάρος πχ. tfidf. Δοκιμάστε τις προτεινόμενες λύσεις ή προτείνετε μια δική σας προσέγγιση στο πρόβλημα.

Υλοποίηση

Η υλοποίηση της μηχανής αναζήτησης *προτείνεται* να πραγματοποιηθεί με χρήση `Java`, `Lucene` και `DL4J`. Μπορείτε να δοκιμάσετε μια άλλη μηχανή αναζήτησης (πχ. `ElasticSearch`, `Solr`) και άλλη γλώσσα προγραμματισμού (πχ. `python`), αλλά πιθανόν να έχετε περιορισμένη υποστήριξη από τη διδάσκουσα.

Εργαλεία: τα εργαλεία που θα χρειαστείτε μπορείτε να τα βρείτε παρακάτω

Lucene	https://lucene.apache.org/
	https://lucene.apache.org/core/downloads.html - κατεβάστε από εδώ την τελευταία έκδοση της σειράς 7.x όχι της 8.x.
DL4J	https://deeplearning4j.org/
	Παράδειγμα υλοποίησης word2vec με χρήστη DL4J:
	https://jrmerwin.github.io/deeplearning4j-docs/programmingguide/07_nlp
Java 8++	
trec_eval	https://trec.nist.gov/trec_eval/ (διαθέσιμο στο eclass)
WordNet	https://wordnet.princeton.edu/download/current-version (διαθέσιμο στο eclass)

Συλλογή `IR2020`:

Διαθέσιμη στο eclass στο φάκελο «Προγραμματιστική εργασία\Συλλογή IR2020»

Βιβλιογραφία - Πηγές:

Mikolov, T., Corrado, G., Chen, K., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (ICLR 2013), 1–12.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS, 1–9.

Ημερομηνίες υποβολής φάσεων εργασίας (ενδέχεται να τροποποιηθούν)

Φάση 1: 22 Απριλίου 2020

Φάση 2: 24 Μαΐου 2020

Φάση 3: ημ/νία εξέτασης μαθήματος ή τέλος εξαμήνου