Αναζήτηση και Συσταδοποίηση διανυσμάτων στη C++

Εργασία 1

ΤΣΟΥΤΣΑΝΗΣ ΑΡΙΣΤΟΤΕΛΗΣ - 1115201600185
ΧΙΟΝΙΔΗΣ ΑΛΕΞΑΝΔΡΟΣ - 1115201400226

MAKEFILE:

Εντολή make για compile όλων τον εκτελέσιμων.

Εντολή make clean για διαγραφή όλων τον εκτελέσιμων, των linked objects αρχείων και των .txt αρχείων.

LSH:

./lsh -d <train-file> -q <query-file> -k <#hash functions> -L <#hashtables> -o <output file> -R <range search> -N <#nearest neighbors> -t <#images for sampling>

Οι παραπάνω παράμετροι μπορούν να δίνονται και κατά την εκτέλεση του προγράμματος από το χρήστη. Η παράμετρος -t είναι προαιρετική και προσδιορίζει των αριθμό των εικόνων που υπολογίζουν τη μέση απόστασή τους από τον πλησιέστερο γείτονα, ώστε να προσδιοριστεί το w (by default είναι 100).

Το αρχείο lsh.cpp αποτελεί τη main του προγράμματός μας, όπου διαβάζει τις εικόνες του dataset, τα queries από το query file και καλεί τις αντίστοιχες συναρτήσεις για την κατασκευή του LSH και την παραγωγή κι εκτύπωση αποτελεσμάτων.

Το αρχείο structs.cpp περιέχει όλες τις δομές που χρησιμοποιούνται για τη κατασκευή του LSH και την εύρεση των neighbors (nearest ή in range). Η κλάση Hash χρησιμοποιείται για τη κατασκευή ενός hash table. Συγκεκριμένα, αποθηκεύονται οι συναρτήσεις κατακερματισμού (h), η g συνάρτηση κατακερματισμού καθώς και ένας πίνακας με τα hash table buckets τύπου Hash_list. Η κλάση Hash_list αποτελεί το bucket του hash table και περιέχει μια λίστα (vector<>) με τις εικόνες του, καθώς και τα αντίστοιχα labels που υποδηλώνουν τις αντίστοιχες g από τις οποίες προήλθαν. Η κλάση PQ περιέχει μία ουρά προτεραιότητας, όπου κατά την κατασκευή της βρίσκει τους N πλησιέστερους γείτονες και τους αποθηκεύει. Επίσης περιέχει τις συναρτήσεις για το range search.

Το αρχείο hash_functions.cpp περιέχει τη κλάση Hash_Function η οποία αποτελείται από τα στοιχεία που χρειάζονται για την κατασκευή μιας συνάρτησης κατακερματισμού (h) και τις αντίστοιχες συναρτήσεις.

Το αρχείο helping_functions.cpp περιέχει διάφορες βοηθητικές συναρτήσεις για την υλοποίηση του LSH (και των άλλων εκτελέσιμων).

HyperCube:

./cube -d <train-file> -q <query-file> -k <projected dimension> -M <#possible neighbors> -o <output file> -R <range search> -N <#nearest neighbors> -t <#images for sampling> -probes <#vertices to check>

Οι παραπάνω παράμετροι μπορούν να δίνονται και κατά την εκτέλεση του προγράμματος από το χρήστη. Η παράμετρος -t είναι προαιρετική και προσδιορίζει των αριθμό των εικόνων που υπολογίζουν τη μέση απόστασή τους από τον πλησιέστερο γείτονα, ώστε να προσδιοριστεί το w (by default είναι 100).

Το αρχείο cube.cpp αποτελεί τη main του προγράμματός μας, όπου διαβάζει τις εικόνες του dataset, τα queries από το query file και καλεί τις αντίστοιχες συναρτήσεις για την κατασκευή του HyperCube και την παραγωγή κι εκτύπωση αποτελεσμάτων.

Το αρχείο structs.cpp περιέχει όλες τις δομές που χρησιμοποιούνται για την κατασκευή του HyperCube και την εύρεση των neighbors (nearest ή in range). Η κλάση Cube χρησιμοποιείται για την κατασκευή του HyperCube. Συγκεκριμένα, αποθηκεύονται οι συναρτήσεις κατακερματισμού (h) και η f στην οποία κάνουν cointoss, η g συνάρτηση κατακερματισμού(κορυφή), καθώς και ένας πίνακας με τις κορυφές του υπερκύβου τύπου Hash_list. Η κλάση PQ περιέχει μία ουρά προτεραιότητας, όπου κατά την κατασκευή της βρίσκει τους N πλησιέστερους γείτονες και τους αποθηκεύει. Επίσης περιέχει τις συναρτήσεις για το range search.

Το αρχείο hash_functions.cpp περιέχει τη κλάση Hash_Function η οποία αποτελείται από τα στοιχεία που χρειάζονται για την κατασκευή μιας συνάρτησης κατακερματισμού (h) και τις αντίστοιχες συναρτήσεις.

Το αρχείο helping_functions.cpp περιέχει διάφορες βοηθητικές συναρτήσεις για την υλοποίηση του HyperCube (και των άλλων εκτελέσιμων). Όπως η συνάρτηση get_route η οποία, αναλόγως τη διάσταση στην οποία

προβάλουμε, επιστρέφει τη διαδρομή που πρέπει να ακολουθήσουμε για να επισκεφτούμε γείτονες κατά αύξουσα hamming distance.

Clustering:

./cube -i <train-file> -c <configuration file> -m <classic or lsh or hypercube> -o <output file> -complete <yes or YES> -t <#images for sampling>

Οι παραπάνω παράμετροι μπορούν να δίνονται και κατά την εκτέλεση του προγράμματος από το χρήστη. Η παράμετρος -t είναι προαιρετική και προσδιορίζει τον αριθμό των εικόνων που υπολογίζουν τη μέση απόστασή τους από τον πλησιέστερο γείτονα, ώστε να προσδιοριστεί το w (by default είναι 100). Η παράμετρος -complete είναι προαιρετική και προσδιορίζει αν στο output θα υπάρχει η πλήρης λίστα εικόνων για κάθε cluster ή όχι (-complete yes/YES για πλήρη εκτύπωση).

Το αρχείο cluster.cpp αποτελεί τη main του προγράμματός μας, όπου διαβάζει τις εικόνες του dataset και το configuration file με τις παραμέτρους (αν δεν δοθεί το k στο αρχείο αυτό, αναμένει από το χρήστη να το δώσει). Ανάλογα με το -method που επιλέχθηκε, υπάρχει διαφορετική περίπτωση και υλοποιείται ο αντίστοιχος αλγόριθμος, καλώντας τις κατάλληλες συναρτήσεις.

Το αρχείο structs_cluster.cpp περιέχει τις δομές που χρειάζονται για την συσταδοποίηση. Η κλάση KMeans περιέχει τα Clusters και τις συναρτήσεις που χρειάζονται για την αρχικοποίηση και τη δημιουργία τους, την ενημέρωση (διαφορετική για κάθε method), καθώς και τον υπολογισμό της σιλουέτας. Η κλάση Cluster περιέχει τα images και το κεντροειδές της, καθώς και την υλοποίηση του αλγορίθμου kmedians++ για την ενημέρωση του κεντροειδούς, μαζί με τις υπόλοιπες βοηθητικές συναρτήσεις της. Η κλάση Point είναι βοηθητική για το Cluster, η οποία περιέχει το image και το cluster id που ανήκει.

Τα αρχεία structs.cpp και hash_functions.cpp περιέχουν κλάσεις και συναρτήσεις που χρειάζονται για τον LSH και HyperCube αντίστοιχα.

Το αρχείο helping_functions.cpp περιέχει διάφορες βοηθητικές συναρτήσεις για την υλοποίηση του clustering καθώς και την υλοποίηση του LSH ή του Hypercube αν δεν έχει δοθεί η κλασική μέθοδος.

Παραδοχές:

Η υλοποίηση μας έχει ακολουθήσει κατα γράμμα την εκφώνηση της εργασίας, καθώς και τη θεωρία από τις διαφάνειες Αξίζει να σημειωθεί όμως ότι:

- Κατά το διάβασμα του image, προσθέτουμε 3 bytes στο τέλος του διανύσματος που υποδηλώνουν τη θέση του image στο αρχικό dataset. (Θεωρούμε ότι δεν θα διαβάσουμε περισσότερα από ~16εκ. images)
- Έχουμε προσθέσει στο command line το προαιρετικό flag -t, το οποίο αφορά τον αριθμό των samples της δειγματοληψίας για τον υπολογισμό της μέσης απόστασης με τον πλησιέστερο γείτονα και κατ' επέκταση του w.
- Έχουμε δώσει έμφαση στη μείωση του χρόνου εκτέλεσης των προγραμμάτων μας, ιδιαίτερα των προσεγγιστικών αλγορίθμων.
- Για τον υπολογισμό του $a_i = (p_i - s_i)/w$, όπου $a_i \in \mathbb{Z}$, επειδή το $s_i \leq w$και $p_i \in [0,255]$και w≃40.000, το a_iμπορεί να πάρει τιμή -1 ή 0 κι εξαρτάται αποκλειστικά από το $p_i - s_i$, δηλαδή αν $p_i - s_i < 0$τότε $a_i = -1$, ενώ αν $p_i - s_i \geq 0$τότε $a_i = 0$. Συνεπώς η συνάρτηση υπολογισμού του a_i έχει απλοποιηθεί στον κώδικα μας με βάση τον παραπάνω τύπο.
- Μετά από πειραματισμό, ορίζουμε το w ως το τριπλάσιο της μέσης απόστασης ανάμεσα σε πλησιέστερους γείτονες, καθώς έχουμε τη καλύτερη αναλογία χρόνου-ακρίβειας.