Κ23γ: Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα Χειμερινό εξάμηνο 2020-21
1η Προγραμματιστική Εργασία
Αναζήτηση και συσταδοποίηση διανυσμάτων στη C/C++

Παναγιώτης - Άγγελος Ίσηρης Α.Μ. 1115 2015 00233
Πέτρος - Φώτης Καμπέρη Α.Μ. 1115 2017 00043

Περιγραφή προγράμματος

Α ερώτημα:

Καταρχάς, υλοποιήσαμε μια κλάση Point η οποία διαμορφώθηκε με σκοπό να αποθηκεύει διανύσματα, και να τους δίνει ένα αναγνωριστικό id. Έπειτα ορίσαμε μια κλάση H η οποία έχει ως σκοπό την υλοποίηση των συναρτήσεων h της LSH, καθώς και μία κλάση G η οποία ομοίως υλοποιεί τις συναρτήσεις g της LSH. Ακολούθως, για την υλοποίηση του Locality Sensitive Hashing (LSH) βάσει της μετρικής L1 (manhattan distance) κατασκευάσαμε την κλάση LSH η οποία περιέχει ένα vector με αντικείμενα της κλάσης G, ένα vector από αντικείμενα της κλάσης Point που αναπαριστά το dataset, τις παραμέτρους που χρειαζόμαστε με βάση την θεωρία και την δομή του LSH.

Ειδικότερα, η δομή του LSH αποτελείται από έναν δισδιάστατο πίνακα που περιέχει vector από αντικείμενα της κλάσης Point. Ο πίνακας αποτελείται από L γραμμές όσες και οι συναρτήσεις g και k στήλες όσες και οι συναρτήσεις h που χρησιμοποιεί κάθε συνάρτηση g. Ουσιαστικά, κάθε γραμμή του πίνακα είναι ένα hash table και κάθε στοιχείο αυτής αποτελεί ένα κάδο.

Επιπλέον στην κλάση της LSH δημιουργήθηκαν 3 συναρτήσεις. Οι 2 εξ αυτών έχουν ως σκοπό την εύρεση γειτονικών σημείων από ένα άλλο τα οποία έχουν παρθεί από το dataset που δέχεται ως όρισμα η LSH. Η 3η συνάρτηση έχει σκοπό την εκτύπωση των αποτελεσμάτων των 2 προηγούμενων σε ένα αρχείο με μια προκαθορισμένη μορφή όπως δόθηκε στην εκφώνηση, καθώς και τον υπολογισμό του χρόνου εκτέλεσης της εκάστοτε διαδικασίας.

Όσον αφορά στην τυχαία προβολή στον υπερκύβο ακολουθήσαμε παρόμοια διαδικασία με την LSH με την διαφορά ότι, η κλάση F αντικαθιστά την θέση της κλάσης G. Για την υλοποίηση αυτή δημιουργήσαμε την κλάση cube η οποία περιέχει ένα vector με αντικείμενα της κλάσης F, ένα vector από αντικείμενα της κλάσης Point που αναπαριστά το dataset, τις παραμέτρους που χρειαζόμαστε με βάση την θεωρία και την δομή του υπερκύβου.

Ειδικότερα, η δομή του υπερκύβου αποτελείται από ένα unordered_map<long int, vector<Point>> οπού κάθε κλειδί αντιστοιχίζεται σε ένα vector από αντικείμενα Point, το οποίο αναπαριστά ένα κάδο.

Επίσης, στην κλάση της Cube δημιουργήθηκαν ομοίως 3 συναρτήσεις. Οι 2 εξ αυτών έχουν ως σκοπό την εύρεση γειτονικών σημείων από ένα άλλο τα οποία έχουν παρθεί από το dataset που δέχεται ως όρισμα η Cube. Η 3η συνάρτηση έχει σκοπό την εκτύπωση των αποτελεσμάτων των 2 προηγούμενων σε ένα αρχείο με μια προκαθορισμένη μορφή όπως δόθηκε στην εκφώνηση, καθώς και τον υπολογισμό του χρόνου εκτέλεσης της εκάστοτε διαδικασίας.

Β ερώτημα:

Για το ερώτημα αυτό υλοποιήσαμε 4 κλάσεις. Η πρώτη από αυτές η κλάση Cluster κληρονομείται από τις υπόλοιπες τρεις και περιέχει ένα vector από αντικείμενα Point που αντιστοιχεί στο dataset, ένα vector από αντικείμενα point που αποθηκεύονται τα κεντροειδή που αρχικοποιεί με την μέθοδο kmeans++, έναν ακέραιο για το πλήθος τους, μια συνάρτηση που αφορά την εκτύπωση των ζητούμενων αποτελεσμάτων με την προκαθορισμένη μορφή που ζητήθηκε, καθώς και την συνάρτηση αξιολόγησης Silhouette.

Οι υπόλοιπες 3 κλάσεις αφορούν τις 3 τεχνικές clustering που ζητούνται από την εκφώνηση. (Reverse Range Search for LSH, Reverse Range Search for Hypercube, Lloyds). Και οι 3 κλάσεις αυτές αποθηκεύουν τα cluster τους σε δομή unordered_map<int, vector<Point>>, όπου το κλειδί είναι το id του εκάστοτε cluster.

Αρχεία και περιγραφή:

- utilities.hpp → Δηλώσεις class Point, class H, class G και απαραίτητες δομές και συναρτήσεις, καθώς και συναρτήσεις διαβάσματος αρχείων.
- utilities.cpp → υπολοποίηση όσων έχουν δηλωθεί στο utilities.hpp
- lsh.hpp → Δηλώση class LSH
- lsh.cpp → Υλοποίηση class LSH και λειτουργιών της
- cube.hpp → Δηλώσεις class F και class Cube
- cube.cpp → Υλοποίηση των δηλώσεων του αρχείου cube.hpp
- main_lsh.cpp → Είσοδος και επεξεργασία ορισμάτων και κλήση των των λειτουργιών lsh.cpp .
- main_cube.cpp → Είσοδος και επεξεργασία ορισμάτων και κλήση των των λειτουργιών cube.cpp .
- clusterkmeans.hpp → Δήλωση των κλάσεων Cluster, Lloyds, Rev_Lsh, Rev_Cube και οι απαραίτητες δομές τους.
- clusterkmeans.cpp → Υλοποίηση των δηλώσεων του αρχείου clusterkmeans.hpp
- clustermain.cpp → Είσοδος και επεξεργασία ορισμάτων και κλήση των των λειτουργιών clusterkmeans.cpp .

- Makefile → Το αρχείο για την μεταγλώττιση του προγράμματος.

Μεταγλώττιση και εκτέλεση προγραμμάτων:

Α ερώτημα:

make lsh

./lsh –d train-images-idx3-ubyte –q t10k-images-idx3-ubyte –k 4 -L 5 -o outputlsh.txt -N 1 -R 10000

make cube

./cube –d train-images-idx3-ubyte –q t10k-images-idx3-ubyte –k 14 -M 10 -probes 2 -o outputcube.txt -N 1 -R 10000

Β ερώτημα:

make cluster

./cluster –i train-images-idx3-ubyte –c cluster.conf -o outputcluster.txt -complete -m lloyds

ή

./cluster –i train-images-idx3-ubyte –c cluster.conf -o outputcluster.txt -complete -m Range_Search_LSH

ή

./cluster –i train-images-idx3-ubyte –c cluster.conf -o outputcluster.txt -complete -m Range_Search_Hypercube

(αναλόγως ποια μέθοδο θέλουμε να χρησιμοποιήσουμε!!)

Πηγές από το διαδίκτυο:

Για το διάβασμα του dataset:
https://compvisionlab.wordpress.com/2014/01/01/c-code-for-reading-mnist-data-set/

Για το hamming distance:
https://en.wikipedia.org/wiki/Hamming_distance

Για το uniform distribution:
https://en.cppreference.com/w/cpp/numeric/random/uniform_int_distribution

Σχόλια - διευκρινίσεις:

Α ερώτημα:

lsh (ολοκληρώθηκε)

hypercube(ολοκληρώθηκε)

Β ερώτημα:

Lloyd (ολοκληρώθηκε)

Range search lsh (ολοκληρώθηκε - δεν έχουμε τα επιθυμητά αποτελέσματα)

Range search cube (ολοκληρώθηκε - δεν έχουμε τα επιθυμητά αποτελέσματα)