ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΑΛΓΟΡΙΘΜΙΚΑ ΠΡΟΒΛΗΜΑΤΑ

1Η ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ
ΑΝΑΖΗΤΗΣΗ ΚΑΙ ΣΥΣΤΑΔΟΠΟΙΗΣΗ ΔΙΑΝΥΣΜΑΤΩΝ

Μέλη ομάδας: Δημήτριος Σταύρος Κωστής (ΑΜ: 1115201700304)
Ορέστης Θεοδώρου (ΑΜ: 1115202000058)

Στα παρακάτω αρχεία υλοποιείται η δημιουργία αλγορίθμου LSH δομής υπερκύβου για την εύρεση πλησιέστερων γειτόνων, όπως και η υλοποίηση αλγορίθμου Kmeans για εύρεση συστάδων.

Αλγόριθμος LSH

Έχει υλοποιηθεί ο αλγόριθμος LSH στα παρακάτω αρχεία:

LSHmain.cpp:

Το αρχείο περιέχει την main συνάρτηση η οποία με βάση τις μεταβλητές εισόδου, δημιουργεί τα LSH hash table με βάση τα mnist images, εισάγει τα queries και βρίσκει τους κοντινότερους γείτονες τους. Επιπλέον, καλεί συναρτήσεις για την εύρεση των κοντινότερων γειτόνων με BruteForce. Τέλος, καλεί τις συναρτήσεις εξόδου για την σύνταξη του αρχείου εξόδου με τις ανάλογες πληροφορίες.

ImageVector.cpp και ImageVector.h:

Η κλάση ImageVector αναπαριστά το διάνυσμα των mnist images και περιέχει επιπρόσθετες πληροφορίες για την εκάστοτε εικόνα και τις ανάλογες συναρτήσεις που βοηθούν στην υλοποίηση του LSH.
Οι συναρτήσεις της κλάσης διαχειρίζονται τα στοιχεία neighbours, στα οποία είναι αποθηκευμένες πληροφορίες για τα data points που αναπαριστούν τους πλησιέστερους γείτονες.

HashTableLSH.cpp και HashTableLSH.h:

Η κλάση Hash_Table_LSH υλοποιεί τον αλγόριθμο LSH.

Functions.cpp και Functions.h:

Στα αρχεία αυτά περιέχονται οι παρακάτω συναρτήσεις:

- Την calculateEuclideanDistance που βρίσκει την απόσταση μεταξύ δυο ImageVectors, που βρίσκει την ευκλείδεια απόσταση.
- Την CreateHashTables που δημιουργεί τα LSH table (δίνουμε δικές μας τιμές για το w).
- Η συνάρτηση readImages λαμβάνει το όνομα ενός αρχείου, με προδιαγραφές dataset MINST και εισάγει το κάθε datapoint σε ένα ImageVector και το επιστρέφει.
- Η συνάρτηση readQuery πραγματοποιεί αντίστοιχη διαδικασία με την readImage, διαβάζει το query με προδιαγραφές dataset MINST και επιστρέφει το κατάλληλο imageVector.
- Περιέχει την υλοποίηση της BruteForce διαδικασίας εύρεσης του πλησιέστερου γείτονα, η συνάρτηση BruteForce λαμβάνει σαν εισόδους τα queries και τα image, καλεί FindNearestNeighborsBruteForce η οποία για βρίσκει την απόσταση κάθε image από το query.

exitFunctions.cpp και exitFunctions.h:

Τα αρχεία περιέχουν τη δομή ExitFile που αποθηκεύει τις πληροφορίες που θέλουμε στο outputFile.
Η συνάρτηση createFile λαμβάνει τις τελικές πληροφορίες και δημιουργεί το αρχείο εξόδου με βάση τις προδιαγραφές.

comFunctions.cpp και comFunctions.h:

Τα αρχεία περιέχουν τις κατάλληλες δομές για τις πληροφορίες εισόδου που λαμβάνουν τα εκτελέσιμα αρχεία.

HashTableLSH.cpp και HashTableLSH.h:

Στα αρχεία αυτά περιέχονται οι παρακάτω συναρτήσεις:

- NormalDistrubutionSeeder: Δημιουργία τιμών κανονικής κατανομής για εκχώρηση στον πίνακα v[] που χρειάζεται για την δημιουργία των συναρτήσεων h.
- UniformDistributionSeeder: Δημιουργία τιμών κανονικής κατανομής για εκχώρηση στον πίνακα t[] που χρειάζεται για την δημιουργία των συναρτήσεων h.
- calculateH: Υπολογισμός συναρτήσεων h.

- AmplifiedFunctionG: Υπολογισμός index με τη δημιουργία της συνάρτησης G.
- InsertItem: Βρίσκουμε το index και εισάγουμε το data point στο LSH hash table.
- FakeInsert: Βρίσκουμε το index του data point που θα είχε αν κάναμε εισαγωγή, χωρίς όμως να πραγματοποιείται εισαγωγή του στοιχείου.
- FindNeighbours: Βρίσκουμε τους κοντινότερους γείτονες του εκάστοτε query.
- LSHnn: Βρίσκουμε τους κοντινότερους γείτονες χρησιμοποιώντας τη δομή LSH hash table.
- RangeSearch: Βρίσκουμε τους κοντινότερους γείτονες βάση της δοθείσας ακτίνας.

HyperCube.cpp και HyperCube.h

Στα αρχεία αυτά περιέχονται οι παρακάτω συναρτήσεις:

- getIndex: Βρίσκουμε το index.
- CalculateF: Υπολογίζουμε το F για να βρούμε τη κορυφή του κύβου όπου θα καταχωρηθεί το στοιχείο.
- InsertItem: Βρίσκουμε το index και εισάγουμε το data point στον υπερκύβο.
- FakeInsert: Βρίσκουμε το index του data point που θα είχε αν κάναμε εισαγωγή, χωρίς όμως να πραγματοποιείται εισαγωγή του στοιχείου
- FindNeighbours: Βρίσκουμε τους πλησιέστερους γείτονες ελέγχοντας την κορυφή που βρίσκεται το στοιχείο κι άλλες m κορυφές.
- PrintTable: Εκτυπώνουμε τα στοιχεία του κύβου.
- GetProbe: Βρίσκουμε την κορυφή του κύβου που πρέπει να ελέγξουμε υπολογίζοντας την απόσταση hamming.
- RangeSearch: Βρίσκουμε τους κοντινότερους γείτονες βάση της δοθείσας ακτίνας.
- CubeSearch: Βρίσκουμε τους κοντινότερους γείτονες χρησιμοποιώντας τη δομή hypercube.
- BruteForce: Επαναληπτική αναζήτηση στον κύβο.

CubeFunctions.cpp και ClusterFunctions.h

Στα αρχεία αυτά περιέχονται βοηθητικές συναρτήσεις για την υλοποίηση του hypercube.

DictionaryF.cpp και DictionaryF.h

Στα αρχεία αυτά περιέχεται η κλάση με την οποία κάνουμε mapping για το πού εκχωρήθηκε το κάθε στοιχείο.

HypercubeMain.cpp

Στο αρχείο αυτό περιέχεται η κύρια συνάρτηση για την υλοποίηση του hypercube.

Kmeans.cpp και Kmeans.h:

Στα αρχεία αυτά περιέχονται οι παρακάτω συναρτήσεις:

- InitializationKmeansPlusPlus: Υλοποίηση της αρχικοποίησης Kmeans++.
- LloydMethod: Υλοποίηση μεθόδου Lloyd για ανάθεση στοιχείων.
- MacQueenMethod: Υλοποίηση μεθόδου MacQueen. Δεν έχει υλοποιηθεί ολόκληρος.
- PrintClusters: Εκτύπωση συστάδων.
- PrintCentroids: Εκτύπωση κεντροϊδών.

Cluster.cpp και Cluster.h

Στα αρχεία αυτά περιέχεται η κλάση που αναπαριστά μια συστάδα.
Clusterfunctions.cpp και Clusterfunctions.h

Στα αρχεία αυτά περιέχονται βοηθητικές συναρτήσεις του Kmeans.

ClusterMain.cpp

Η κύρια συνάρτηση με την οποία δημιουργείται, αρχικοποιείται και ενημερώνεται ο Kmeans.

MakeFile

Στο αρχείο αυτό περιέχεται η μεταγλώττιση των αρχείων και η δημιουργία των εκτελέσιμων αρχείων lsh, cube, cluster. Τα αρχεία αυτά εκτελούνται βάση του προτύπου που αναφέρεται στην εκφώνηση της εργασίας.

Για μεταγλώττιση: make all
Για clean: make clean

Η υλοποίηση του προγράμματος έχει γίνει με τη χρήση συστήματος διαχείρισης εκδόσεων λογισμικού και συνεργασίας (git).