Gruppo 7

K-D TREE AND NIPALS ALGORITHM IN LINGUAGGIO ASSEMBLY X86-32+SSE E X86-64+AVX

FASI DELLO SVILUPPO

Implementazione in C

Metodi in assembly

Valutazione dei risultati e ottimizzazione

Gruppo 7

Metodi in C

PCA

Teniamo salvato una trasposizione del dataset per poterci facilitare calcoli.
Per risparmiare memoria deallochiamo ogni volta il risultato del calcolo precedente e ciò non impiega molto tempo a livello prestazionale.

K-D TREE

Composto da un algoritmo normale e uno ricorsivo.
Il calcolo del mediano avviene nella maniera più semplice: tramite l'ordinamento.
Utilizziamo un array di indici per tenere traccia degli id dei vicini utili nel Range Query.

RANGE QUERY

Una volta generata l'area H tramite il minimo e il massimo per ogni dimensione, prendiamo due strade diverse in caso la PCA sia stata eseguita o meno.
Tutto ciò avviene con un metodo di supporto che fa i calcoli delle distanze senza preoccuparsi di quale sia l'input specifico.

OTTIMIZZAZIONI

PRINCIPIO DI LOCALITÀ

Ovunque possibile, abbiamo cercato di leggere i dati delle matrici in modo sequenziale, per far si che la possibilità che siano memorizzati in cache aumenta e quindi il tempo di accesso al dato diminuisce. Per esempio abbiamo memorizzato il secondo operando del prodotto di matrici in Column Major Order.

METODI IN ASSEMBLY

Abbiamo selezionato alcuni metodi adatti ad essere riscritti in assembly per poter sfruttare le operazioni vettoriali offerte.

Metodi in Assembly

PRODMATRIX

Il metodo si preoccupa di eseguire il prodotto tra matrici. La seconda matrice viene letta per colonne.

SUBMAT

Calcola la differenza tra il dataset e il prodotto fra il vettore colonna u e il vettore riga v.

DIVIDIMAT

Divide ogni elemento di una matrice per un valore costante.

EUCLIDEANDISTANCE

Algoritmo classico per il calcolo della distanza euclidea.

32 BIT VS 64 BIT (C)

Non ci sono reali differenze tra la versione 32 bit e quella 64 bit. Ogni cambiamento effettuato nella versione 32 bit è stata poi riportata nella versione a 64 bit poiché la logica algoritmica dietro il funzionamento dei metodi è identica.

Gruppo 7

PECULIARITÀ 32 BIT (ASSEMBLY)

Nella versione a 32 bit, avendo a disposizione molti meno registri rispetto alla sua controparte a 64, utilizziamo lo Stack per ”salvare” delle variabili inutilizzate per molto tempo, guadagnando così più registri a disposizione. Utilizziamo inoltre i registri XMM per poter eseguire istruzioni SSE.

PECULIARITÀ 64 BIT (ASSEMBLY)

Nella versione a 64 bit usiamo i registri in più che abbiamo a disposizione e anche istruzioni disponibili solo in AVX come VBROADCASTSS. Utilizziamo anche l’istruzione VPERMF128 nella prima parte della riduzione del contenuto dei registri YMM.

MIGLIORAMENTO DELLE PRESTAZIONI

PCA

Sino a un ordine di grandezza più veloce della versione in C

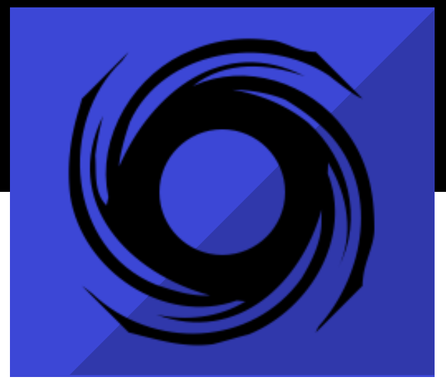
KDTREE

Prestazioni invariate

RANGE QUERY

Impiega fino a metà del tempo necessario alla versione in C

GRAZIE PER L'ATTENZIONE



FRANCESCO PAPALUCA

216639

MATTIA GATTO

216649

ALFREDO SANTO

216637

Gruppo 7