Map Reduce Pseudocode for mat. vector mul. →

Map fn -

① For each element m_{ij} of M do
② product (key, value) pair as $((i,k), (M,j,m_{ij}))$, for $k = 1,2,3,\ldots$ up to the no. of cols of N
③ for each element n_{jk} of N do
④ product (key, value) pairs as $((i,k), (N,j,n_{jk}))$, for $i = 1,2,3,\ldots$ up to the no. of rows of M.
⑤ return set of (key, value) pairs that each key (i,k), has a list of with values (M,j,m_{ij}) & (N,j,n_{jk}) for all possible values of j.

Reduce fn -

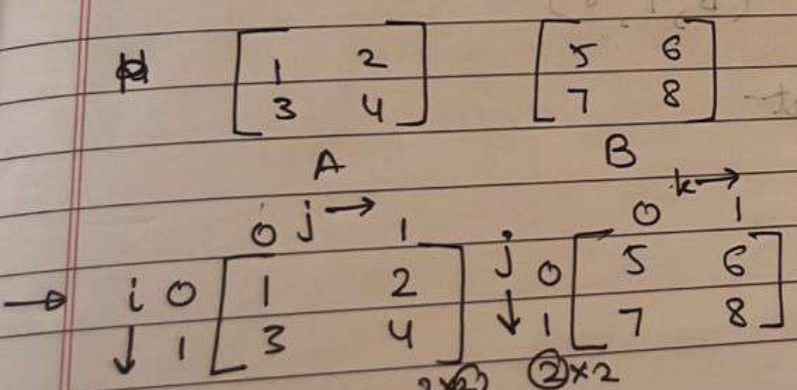
① For each key (i,k) do
② sort value begin with M by j in $list_M$
③ —— " —— " N by j in $list_N$
④ mul. m_{ij} & n_{jk} for jth value of each list
⑤ sum up $m_{ij} * n_{jk}$
⑥ return $\sum_{j=1} m_{ij} * n_{jk}$

Eg: Consider the foll: matrix

$$M = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad N = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

A B

i → row of mat A
j → col. — "
j → row of mat B
k → col. — "



g Gradients: The gating mechanism mitigates vanishing gradients by
nation across long sequences, enabling the model to retain long-term
fectively.

tion in RNNs

ChatGPT can make mistakes. Check important info.

ENG IN 02:01 PM 09-11-2024

① Input file -

Matrix	i/k	j	Value
A	0	0	1
A	0	1	2
A	1	0	3
A	1	1	4
B	0	0	5
B	0	1	6
B	1	0	7
B	1	1	8

Note:
A → i
B → k

② Map fn -

i/k	Value (Matrix, j, Value)
0	(A, 0, 1)
0	(A, 1, 2)
1	(A, 0, 3)
1	(A, 1, 4)
0	(B, 0, 5)
0	(B, 1, 6) ↔ swap
1	(B, 0, 7) ↔ swap
1	(B, 1, 8)

③ ~~Shuffle & Sort~~ -

i/k	Values
0	(A, 0, 1)
0	(A, 1, 2)
1	(A, 0, 3)
1	(A, 1, 4)
0	(B, 0, 5)
0	(B, 0, 7)
1	(B, 1, 6)
1	(B, 1, 8)

③ Shuffle/Group -

(i, k) A B	
(0, 0)	(A,0,1), (A,1,2) (B,0,5), (B,0,7)
(0, 1)	(A,0,1), (A,1,2) (B,1,6), (B,1,8)
(1, 0)	(A,0,3), (A,1,4) (B,1,6), (B,1,8)
(1, 1)	(A,0,3), (A,1,4) (B,0,5), (B,0,7)

RNNe

ChatGPT can make mistakes. Check important info.

ENG IN 02:02 PM 09-11-2024

④ Reduce -

(0, 0) $1 \times 5 + 2 \times 7 = 19$

(0, 1) $1 \times 6 + 2 \times 8 = 22$

(1, 1) $3 \times 6 + 4 \times 8 = 50$

(1, 0) $3 \times 5 + 4 \times 7 = 43$

⑤ Hence resultant matrix is → $\begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$

* MapReduce pseudocode for word count problem →

Map phase :

~~map(key, value):~~

map(key, line) :

// Split line into words

words = line.split(" ")

// For each word, emit(word, 1)

for each word in words:

emit(word, 1)

Reduce phase -

reduce(word, counts):

// Sum all counts for word

total_count = sum(counts)

// Emit (word, total_count)

emit (word, total_count)

Doc: "This is apple. Apple is red in color."
Goal: count occurrence of each word in doc. using mapreduce

(#) Doc. Preprocessing - I/P splitting
I/p doc. is split into lines & tokenized by spaces.

① Map Phase:
Mapper reads each line of doc, splits into words & emits each word with count of 1.
O/P -

word	Output: Emitted key-value pair
"this"	("this", 1)
"is"	("is", 1)
"an"	("an", 1)
"apple"	("apple", 1)
"Apple"	("apple", 1)
"is"	("is", 1)
"red"	("red", 1)
"in"	("in", 1)
"color"	("color", 1)

② Shuffle & Sort Phase:
MapReduce automatically groups all values by key, preparing them for reduce phase. All occurrences of particular word will be grouped together.

Intermediate O/P (grouped by key):

("this", [1])
("is", [1][1])
("an", [1])
("apple", [1], [1])
("red", [1])
("in", [1])
("color", [1])

in RNNs

ChatGPT can make mistakes. Check important info.

ENG IN 02:02 PM 09-11-2024

(3) Reduce phase:
Take O/P of map fn (key-value pairs) as i/p & groups them by key & process them to give final O/P.

O/P -
("this", 1)
("is", 1)
("an", 1)
("apple", 2)
("red", 1)
("in", 1)
("color", 1)

Final Result -
After reduce phase word count for doc
O/P
this: 1
is : 2
an : 1
apple: 2
red : 1
in : 1
color: 1