Pyred

```
from collections import defaultdict
from multiprocessing import Pool
import re

def map_function(line):
    word_counts = defaultdict(int)
    words = re.findall(r'\w+', line.lower())
    for word in words:
        word_counts[word] += 1
    return word_counts

def reduce_function(results):
    final_counts = defaultdict(int)
    for word_count in results:
        for word, count in word_count.items():
            final_counts[word] += count
    return final_counts

def mapreduce(filename):
    with open(filename, 'r') as file:
        lines = file.readlines()

    with Pool() as pool:
        mapped_results = pool.map(map_function, lines)
        reduced_result = reduce_function(mapped_results)

    return reduced_result

if __name__ == "__main__":
    filename = 'your_book_dataset.txt'
    word_counts = mapreduce(filename)
    for word, count in sorted(word_counts.items(), key=lambda x: x[1], reverse=True)[:10]:
        print(f'{word}: {count}')
```