README.md

AWS-EMR-MapReduce-Hebrew-3gram-deleted-estimation

AWS/Hadoop Elastic Map Reduce application to calculate deleted estimation probability on hebrew 3grams

Project GitHub repository link: https://github.com/itaybou/AWS-Hadoop-EMR-MapReduce-Hebrew-3gram-deleted-estimation

Hebrew 3gram S3 path: **s3n://datasets.elasticmapreduce/ngrams/books/20090715/heb-all/3gram/data**

Created by:

Itay Bouganim : 305278384

Sahar Vaya : 205583453

Table of contents

General info

In this assignment you will generate a knowledge-base for Hebrew word-prediction system, based on Google 3-Gram Hebrew dataset, using Amazon Elastic Map-Reduce (EMR). The produced knowledge-base indicates the probability of each word trigram found in the corpus. The project goal is to predict the most probable third word for every hebrew 2 word combination by using the deleted estimation method.

Deleted estimation method

Deleted estimation method is a held out method. The deleted estimation method, uses a form of two-way cross validation, as follows:

$$P_{del}(w_1 \cdots w_n) = \frac{T_r^{01} + T_r^{10}}{N(N_r^0 + N_r^1)} \; where \; C(w_1 \cdots w_n) = r$$

Where:

- N is the number of n-grams in the whole corpus.
- Nr0 is the number of n-grams occuring r times in the first part of the corpus.
- Tr01 is the total number of those n-grams from the first part (those of Nr0) in the second part of the corpus.
- Nr1 is the number of n-grams occuring r times in the second part of the corpus.
- Tr10 is the total number of those n-grams from the second part (those of Nr1) in the first part of the corpus.

Additonal information

EC2 instances used: Workers -

- Machine types - (64-bit x86) type: M4_LARGE

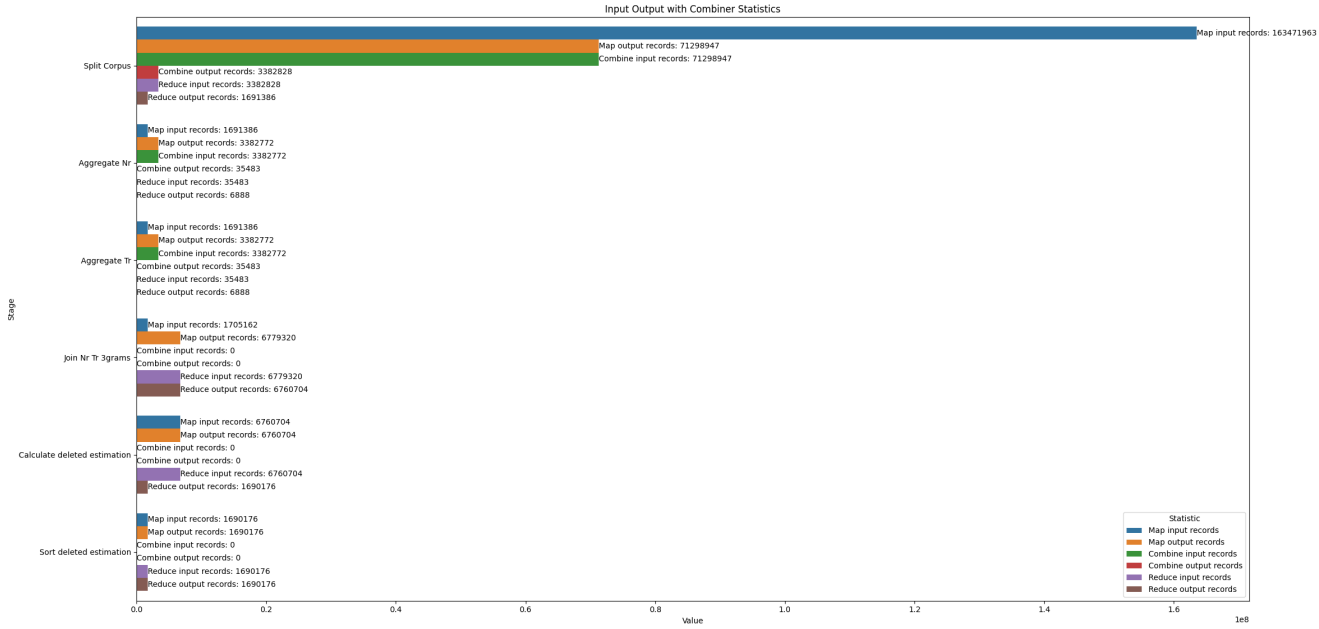
Statistics:

Using the python script in the statistics directory the following statstics charts were produced from the output log-files:

Total ngrams read from corpus: *245866641*

Input Output Records Statistics:

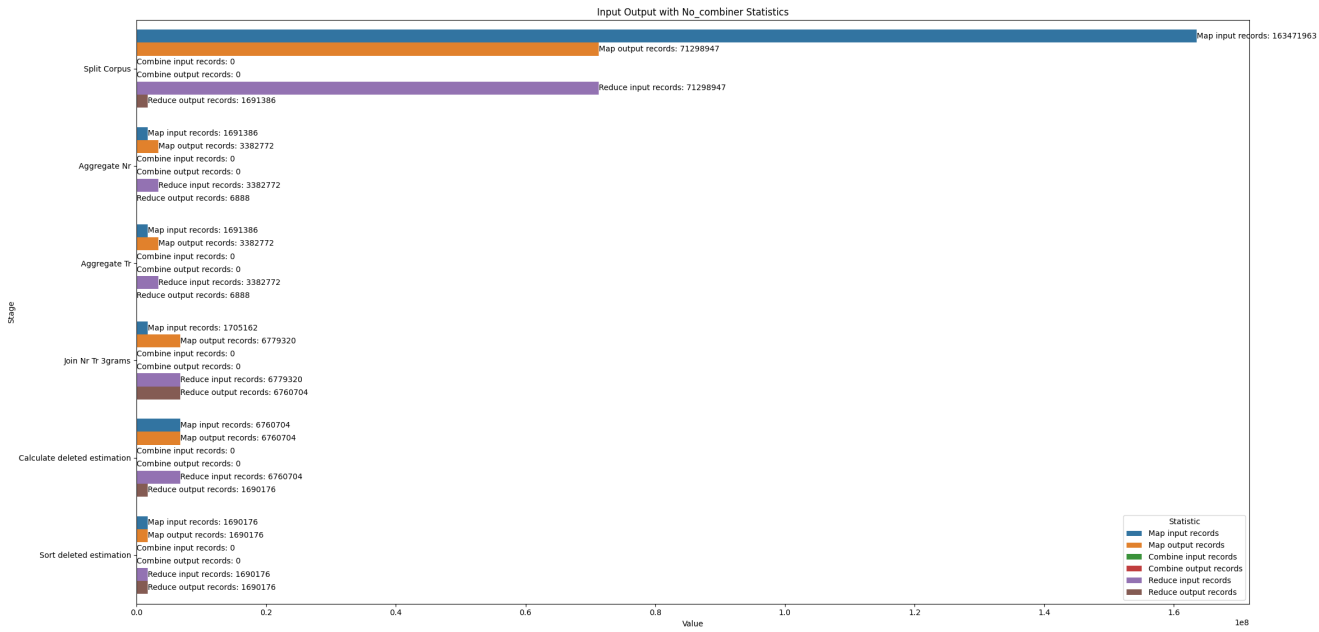
With local aggregation using a Combiner:



	Status	Statistic	Stage	Value
0	Combiner	Map input records	Split Corpus	163471963
1	Combiner	Map output records	Split Corpus	71298947
2	Combiner	Combine input records	Split Corpus	71298947
3	Combiner	Combine output records	Split Corpus	3382828
4	Combiner	Reduce input records	Split Corpus	3382828
5	Combiner	Reduce output records	Split Corpus	1691386
6	Combiner	Map input records	Aggregate Nr	1691386
7	Combiner	Map output records	Aggregate Nr	3382772
8	Combiner	Combine input records	Aggregate Nr	3382772
9	Combiner	Combine output records	Aggregate Nr	35483
10	Combiner	Reduce input records	Aggregate Nr	35483
11	Combiner	Reduce output records	Aggregate Nr	6888
12	Combiner	Map input records	Aggregate Tr	1691386
13	Combiner	Map output records	Aggregate Tr	3382772
14	Combiner	Combine input records	Aggregate Tr	3382772
15	Combiner	Combine output records	Aggregate Tr	35483
16	Combiner	Reduce input records	Aggregate Tr	35483

	Status	Statistic	Stage	Value
17	Combiner	Reduce output records	Aggregate Tr	6888
18	Combiner	Map input records	Join Nr Tr 3grams	1705162
19	Combiner	Map output records	Join Nr Tr 3grams	6779320
20	Combiner	Combine input records	Join Nr Tr 3grams	0
21	Combiner	Combine output records	Join Nr Tr 3grams	0
22	Combiner	Reduce input records	Join Nr Tr 3grams	6779320
23	Combiner	Reduce output records	Join Nr Tr 3grams	6760704
24	Combiner	Map input records	Calculate deleted estimation	6760704
25	Combiner	Map output records	Calculate deleted estimation	6760704
26	Combiner	Combine input records	Calculate deleted estimation	0
27	Combiner	Combine output records	Calculate deleted estimation	0
28	Combiner	Reduce input records	Calculate deleted estimation	6760704
29	Combiner	Reduce output records	Calculate deleted estimation	1690176
30	Combiner	Map input records	Sort deleted estimation	1690176
31	Combiner	Map output records	Sort deleted estimation	1690176
32	Combiner	Combine input records	Sort deleted estimation	0
33	Combiner	Combine output records	Sort deleted estimation	0
34	Combiner	Reduce input records	Sort deleted estimation	1690176
35	Combiner	Reduce output records	Sort deleted estimation	1690176

Without local aggregation:

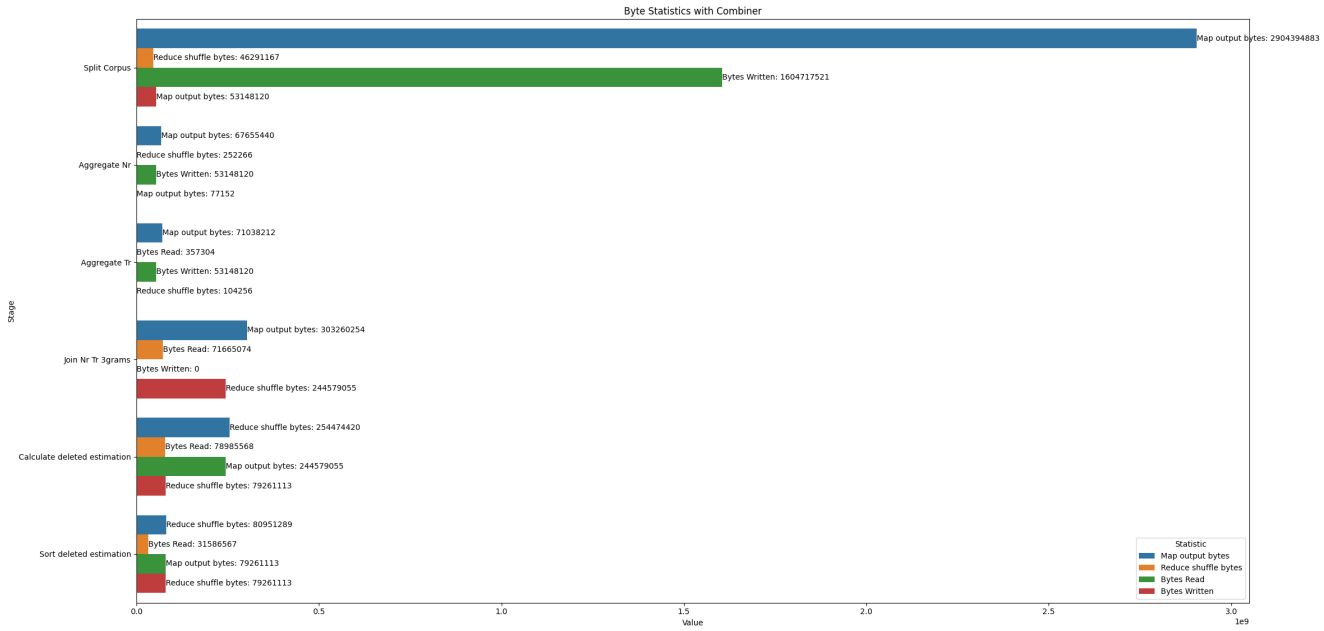


	Status	Statistic	Stage	Value
0	No_combiner	Map input records	Split Corpus	163471963

	Status	Statistic	Stage	Value
1	No_combiner	Map output records	Split Corpus	71298947
2	No_combiner	Combine input records	Split Corpus	0
3	No_combiner	Combine output records	Split Corpus	0
4	No_combiner	Reduce input records	Split Corpus	71298947
5	No_combiner	Reduce output records	Split Corpus	1691386
6	No_combiner	Map input records	Aggregate Nr	1691386
7	No_combiner	Map output records	Aggregate Nr	3382772
8	No_combiner	Combine input records	Aggregate Nr	0
9	No_combiner	Combine output records	Aggregate Nr	0
10	No_combiner	Reduce input records	Aggregate Nr	3382772
11	No_combiner	Reduce output records	Aggregate Nr	6888
12	No_combiner	Map input records	Aggregate Tr	1691386
13	No_combiner	Map output records	Aggregate Tr	3382772
14	No_combiner	Combine input records	Aggregate Tr	0
15	No_combiner	Combine output records	Aggregate Tr	0
16	No_combiner	Reduce input records	Aggregate Tr	3382772
17	No_combiner	Reduce output records	Aggregate Tr	6888
18	No_combiner	Map input records	Join Nr Tr 3grams	1705162
19	No_combiner	Map output records	Join Nr Tr 3grams	6779320
20	No_combiner	Combine input records	Join Nr Tr 3grams	0
21	No_combiner	Combine output records	Join Nr Tr 3grams	0
22	No_combiner	Reduce input records	Join Nr Tr 3grams	6779320
23	No_combiner	Reduce output records	Join Nr Tr 3grams	6760704
24	No_combiner	Map input records	Calculate deleted estimation	6760704
25	No_combiner	Map output records	Calculate deleted estimation	6760704
26	No_combiner	Combine input records	Calculate deleted estimation	0
27	No_combiner	Combine output records	Calculate deleted estimation	0
28	No_combiner	Reduce input records	Calculate deleted estimation	6760704
29	No_combiner	Reduce output records	Calculate deleted estimation	1690176
30	No_combiner	Map input records	Sort deleted estimation	1690176
31	No_combiner	Map output records	Sort deleted estimation	1690176
32	No_combiner	Combine input records	Sort deleted estimation	0
33	No_combiner	Combine output records	Sort deleted estimation	0
34	No_combiner	Reduce input records	Sort deleted estimation	1690176
35	No_combiner	Reduce output records	Sort deleted estimation	1690176

Bytes Records Statistics:

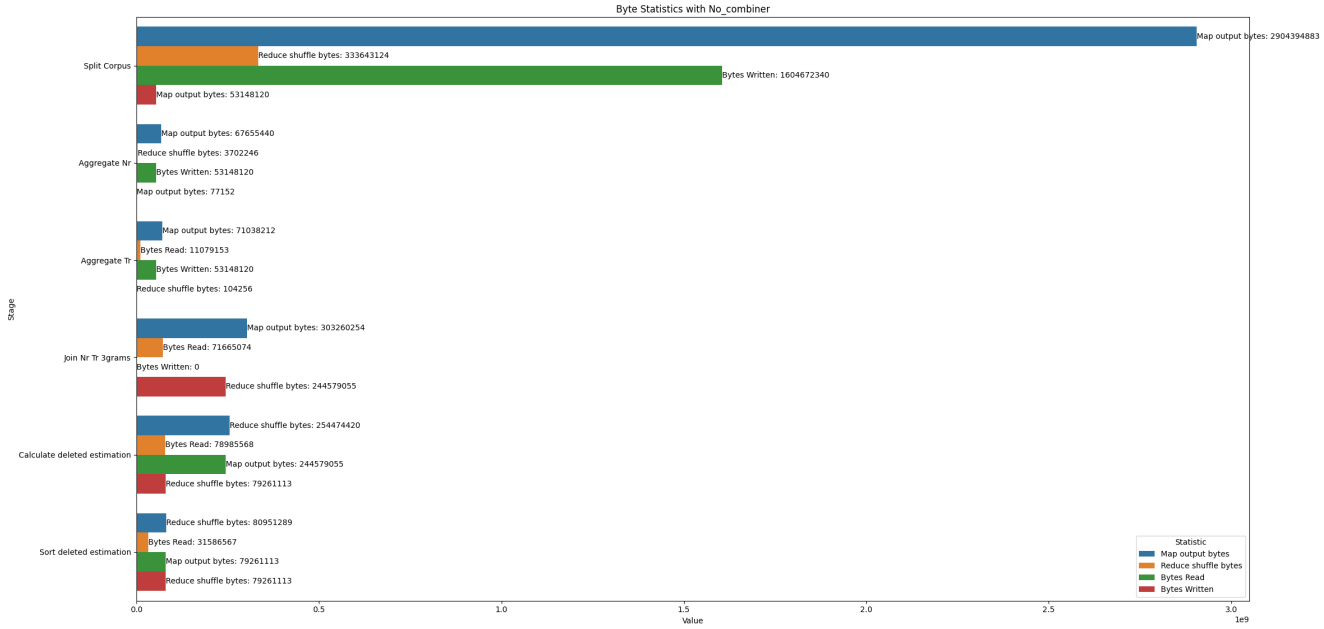
With local aggregation using a Combiner:



	Status	Statistic	Stage	Value
0	Combiner	Map output bytes	Split Corpus	2904394883
1	Combiner	Reduce shuffle bytes	Split Corpus	46291167
2	Combiner	Bytes Read	Split Corpus	1604717521
3	Combiner	Bytes Written	Split Corpus	53148120
4	Combiner	Map output bytes	Aggregate Nr	67655440
5	Combiner	Reduce shuffle bytes	Aggregate Nr	252266
6	Combiner	Bytes Read	Aggregate Nr	53148120
7	Combiner	Bytes Written	Aggregate Nr	77152
8	Combiner	Map output bytes	Aggregate Tr	71038212
9	Combiner	Reduce shuffle bytes	Aggregate Tr	357304
10	Combiner	Bytes Read	Aggregate Tr	53148120
11	Combiner	Bytes Written	Aggregate Tr	104256
12	Combiner	Map output bytes	Join Nr Tr 3grams	303260254
13	Combiner	Reduce shuffle bytes	Join Nr Tr 3grams	71665074
14	Combiner	Bytes Read	Join Nr Tr 3grams	0
15	Combiner	Bytes Written	Join Nr Tr 3grams	244579055
16	Combiner	Map output bytes	Calculate deleted estimation	254474420
17	Combiner	Reduce shuffle bytes	Calculate deleted estimation	78985568
18	Combiner	Bytes Read	Calculate deleted estimation	244579055
19	Combiner	Bytes Written	Calculate deleted estimation	79261113
20	Combiner	Map output bytes	Sort deleted estimation	80951289
21	Combiner	Reduce shuffle bytes	Sort deleted estimation	31586567

	Status	Statistic	Stage	Value
22	Combiner	Bytes Read	Sort deleted estimation	79261113
23	Combiner	Bytes Written	Sort deleted estimation	79261113

Without local aggregation:



	Status	Statistic	Stage	Value
0	No_combiner	Map output bytes	Split Corpus	2904394883
1	No_combiner	Reduce shuffle bytes	Split Corpus	333643124
2	No_combiner	Bytes Read	Split Corpus	1604672340
3	No_combiner	Bytes Written	Split Corpus	53148120
4	No_combiner	Map output bytes	Aggregate Nr	67655440
5	No_combiner	Reduce shuffle bytes	Aggregate Nr	3702246
6	No_combiner	Bytes Read	Aggregate Nr	53148120
7	No_combiner	Bytes Written	Aggregate Nr	77152
8	No_combiner	Map output bytes	Aggregate Tr	71038212
9	No_combiner	Reduce shuffle bytes	Aggregate Tr	11079153
10	No_combiner	Bytes Read	Aggregate Tr	53148120
11	No_combiner	Bytes Written	Aggregate Tr	104256
12	No_combiner	Map output bytes	Join Nr Tr 3grams	303260254
13	No_combiner	Reduce shuffle bytes	Join Nr Tr 3grams	71665074
14	No_combiner	Bytes Read	Join Nr Tr 3grams	0
15	No_combiner	Bytes Written	Join Nr Tr 3grams	244579055
16	No_combiner	Map output bytes	Calculate deleted estimation	254474420
17	No_combiner	Reduce shuffle bytes	Calculate deleted estimation	78985568

	Status	Statistic	Stage	Value
18	No_combiner	Bytes Read	Calculate deleted estimation	244579055
19	No_combiner	Bytes Written	Calculate deleted estimation	79261113
20	No_combiner	Map output bytes	Sort deleted estimation	80951289
21	No_combiner	Reduce shuffle bytes	Sort deleted estimation	31586567
22	No_combiner	Bytes Read	Sort deleted estimation	79261113
23	No_combiner	Bytes Written	Sort deleted estimation	79261113

Word Analysis

2gram	על קבר	אהב את	אולי לא	כבר במאה	מה שהאדם
1 Prediction	על קבר רחל	אהב את יוסף	אולי לא היה	כבר במאה השלישית	מה שהאדם עושה
2 Prediction	על קבר אביו	אהב את המלאכה	אולי לא היתה	כבר במאה העשירית	מה שהאדם הוא
3 Prediction	על קבר שמואל	אהב את זה	אולי לא היו	כבר במאה התשע	מה שהאדם צריך
4 Prediction	על קבר הצדיק	אהב את עשו	אולי לא פחות	כבר במאה הרביעית	מה שהאדם יכול
5 Prediction	על קבר האחים	אהב את אשתו	אולי לא הייתי	כבר במאה התשיעית	מה שהאדם משיג
Decision	We can see that the decision that was made here is correct since the first prediction is a common sentence	We can see that the prediction made here is not as we would expect. we would expect the most common prediction for someone to love his wife and not joseph. The possible reason is that old phrases are weighted the same as newer more updated phrases.	Here the prediction is as we would expect.	Here we would expect a more recent century to appear in in the top 5 predictions.	As we would expect.

2gram	בדיוק באותה	תשובות על	תלויה על	על חלק	ולא תוסיף
1 Prediction	בדיוק באותה מידה	תשובות על שאלות	תלויה על בלימה	על חלק מן	ולא תוסיף קום
2 Prediction	בדיוק באותה צורה	תשובות על השאלות	תלויה על הקיר	על חלק גדול	ולא תוסיף לנפול
3 Prediction	בדיוק באותה שעה	תשובות על כל	תלויה על קיר	על חלק ניכר	ולא תוסיף עוד
4 Prediction	בדיוק באותה דרך	תשובות על בני	תלויה על חוט	על חלק זה	ולא תוסיף לדאבה
5 Prediction	בדיוק באותה תקופה	תשובות על שאלותיו	תלויה על צווארו	על חלק של	ולא תוסיף עצב

2gram	בדיוק באותה	תשובות על	תלויה על	על חלק	ולא תוסיף
Decision	A very good prediction was given in this case.	As we expect all top 5 prediction are about answers to some form of questions	The first prediction is a very common hebrew phrase.	Not as we expected we would expect the 4th prediction to be the first one.	We would expect the 3rd prediction to be the first and the 4th prediction to be switched with the fifth one since the fifth is more common nowdays.

Project workflow and summary

We have only one step that include 5 map-reduce jobs, the following jobs are :

1. **Split corpus** - This map-reduce job firstly filters all the three grams in the corpus by the following regular expression:

   ```
   ^(?:[א-ת]+[א-ת\\d+ ]+|[א-ת]+[א-ת\\d+ ]+[א-ת]|[א-ת]+[א-ת\\d+ ]+[א-ת]+[א-ת\\d+ ]+)$"
   ```

   with this filter we will map only three grams that start with hebrew characters and numbers. The map function will split the corpus logically and not physically by the even and odd line id's, for example, if the input of the map is:

   ```
   1.How are you   1975    400 (400 is the occurences of the three gram in 1975)
   2.How are you   2020    300 (300 is the occurences of the three gram in 2020)
   3.How are you   1985    100 (100 is the occurences of the three gram in 1985)
   ...
   ```

   The output of the map function will be :

   ```
   How are you     1       400,100 (1 indicating that this three gram is in the second part of the corpus)
   How are you     0       300 (1 indicating that this three gram is in the first part of the corpus)
   ...
   ```

   After mapping each three gram to the occurences in each part of the corpus, the reduce function will sum all the occurences in the first and the second part and the reducer output will be :

   ```
   How are you     300     500 (300 is the occurences in the first part of the corpus and 500 is the occurences in
   the second part)
   ...
   ```

2. **Aggregate Nr Tr** - This map-reduce job aggregates the sums of 4 values in the deleted estimation formula: Nr0, Nr1, Tr01, Tr10 by using the input of the splitted corpus:

   ```
   How are you     300     500 (300 is the occurences in the first part of the corpus and 500 is the occurences in
   the second part)
   How are them    300     400 (300 is the occurences in the first part of the corpus and 400 is the occurences in
   the second part)
   ```

   The output of the mapping for the Nr calculation will be:

   ```
   Nr0     300     1,1
   Nr1     500     1
   Nr1     400     1
   ```

   The output of the mapping for the Tr calculation will be:

   ```
   Tr01    300     500,400
   Tr10    500     300
   Tr10    400     300
   ```

After mapping each occurences value to corrosponding Nr/Tr values the reducer will sum the values and the output will be:

- For Nr:

```
Nr0     300     2
Nr1     500     1
Nr1     400     1
```

- For Tr:

```
Tr01    300     900
Tr10    500     300
Tr10    400     300
```

3. **Join Nr Tr with 3grams** - The goal of this job is to join the 3grams with their corrosponding Nr and Tr values. The purpose of the job is to avoid storing a list of 3grams for each occurences value in the previous job, by doing that we remove all local memory usage that is input dependant. This job has two mappers and one reducer. The first mapping function recieves the input from the split corpus job and outputs 4 values for each value as follows:

```
300     THREE_GRAM      How are you     Nr0 (300 is the first split occurences, THREE_GRAM enum indicator and
Nr0 indicator for the first split)
500     THREE_GRAM      How are you     Nr1
300     THREE_GRAM      How are you     Tr01
500     THREE_GRAM      How are you     Tr10
...
```

The second mapping function recieves the input from the aggregate Nr Tr job and maps twice to the following outputs:

- For Nr:

```
300     AGGREGATED      2       Nr0 (300 is the first split occurences, AGGREGATED enum indicator and Nr0
indicator for the first split)
500     AGGREGATED      1       Nr1
400     AGGREGATED      1       Nr1
```

- For Tr:

```
300     AGGREGATED      900     Tr01 (300 is the first split occurences, AGGREGATED enum indicator and Tr01
indicator for the first split)
500     AGGREGATED      300     Tr10
400     AGGREGATED      300     Tr10
```

After mapping each occurences value the reducer will join the occurences with the corrosponding 3grams. The input for the reduces will be sorted by the by the enum indicator and the occurences value and and grouped by the occurences value. The output of the reducer will be:

```
How are you     Nr0     2
How are you     Nr1     1
How are them    Nr1     1
How are them    Nr0     2
How are you     Tr01    900
How are you     Tr10    300
How are them    Tr01    900
How are them    Tr10    400
```

4. **Calculate deleted estimation: (Tr01 + Tr10) / (N * (Nr0 + Nr1))** - The goal of this job is to calculate the deleted estimation value for each 3gram. The mapping function return the 3grams with the operation and aggregated value (same as previous job). The reducer function recieves the total Ngram value N, captures the Nr0, Nr1, Tr01, Tr10 from the mapping function and emits the calculation of the deleted estimation formula. The output will look like:

```
How are you     <probability> (Where probability is in range [0, 1])
How are them    <probability>
```

5. **Sort deleted estimation output** - This goal of this job is to emit the deleted estimation values calculated in the previous in a sorted fashion so that it is sorted by:

    i. The first word of the 3gram.
    ii. If first words are equal sort by second word of the three gram.
    iii. If first 2 words from the 3 gram are equal sort by the probability value. (The output will be sorted by first two words alphabetically ascending and by the probabilities descending)

    If true flag for single file output is given in the `inputs.txt` file (expanded on later in this readme) than the job will use one reducer to output one sorted file. Otherwise, the output will be multiple sorted files.

Local aggregation using Combiners

If true flag for local aggregation is given in the `inputs.txt` file (expanded on later in this readme) than the job will use combiner in order to optimize the redducer job where possible.
The following jobs include an optional Combiner:
Split corpus and **Aggregate Nr Tr**.
Both use the combiner to locally aggregate values (Corpus split aggregation and Nr/Tr values aggragation) before passing them to the reducer.
Statistics for the combiner usage difference can be found in the Statistics section.

The following jobs do not include optional Combiner:

- **Join Nr Tr with 3grams -** Join operation only, No use for combiner.
- **Calculate deleted estimation -** Did not use Combiner in this map reduce job since we perform division operation in order to calculate the deleted estimation probability which is not an associative operation.
- **Sort deleted estimation output -** Sort operation only, no use for combiner.

Setup

1. Install aws cli in your operating system, for more information click here : https://aws.amazon.com/cli/
2. Configure your amazon aws credentials in your .aws directory, alternatively you can set your credentials by using aws cli : write in your cmd - "aws config".

Instructions

1. Inside the project directory compile the project using the command : `mvn package` .
2. Create in the project target directory file named "inputs.txt".
3. Create input bucket and output bucket (u can use the same bucket and create only one bucket) in AWS S3.
4. Put your input bucket, input jar file name (the WordPrediction jar file located in project target directory), output bucket, instace count (the number of EC2 instances you want to run) and true/false value for local aggregation inside the "inputs.txt" file in the following format:

```
<input-bucket> <input-jar-file-name>
<output-bucket>
<worker-instance-count (0 < x < 10)>
<use-local-aggregation (true or false)>
<single-file (true or false)>
```

- `<input-bucket>` - Is the bucket the jar file `<input-jar-file-name>` is located in.
- `<output-bucket>` - Is the bucket the job will store outputs in **(Will be deleted after the job is completed!)**, Can be the same as input bucket.
- `<worker-instance-count (0 < x < 10)>` - The EC2 instance count that will be used for the map-reduce job (value between 0 excluding and 9 including)
- `<use-local-aggregation (true or false)>` - Wether the job will use Combiners for local aggregation in order to lower network overhead.
- `<single-file (true or false)>` - Whether to output single sorted output file (slower) or multiple sorted output files (faster).

5. Make sure your input text file located in the project target directory or in the same directory as the WordPedictionRunner jar file.

6. The application should be run as follows:
   ```
   java -jar WordPedictionRunner.jar
   ```

IMPORTANT NOTES:

- The application automatically uploads the input jar provided in the `inputs.txt` file to the input bucket provided in the `inputs.txt` file.
- When the job is finished the output result and log-files will be automatically downloaded to the directory the `java -jar WordPedictionRunner.jar` was ran from.
- The output bucket provided in the `inputs.txt` file will be automatically deleted.

Examples And Resources

- After compiling the project - project JAR files can be found in the projects target directory.
- Example for the `inputs.txt` text file needed to run the project can be found in the directory of the project.