Guide for Reproducibility of *A Scalable Model for Frequency Distribution of Low Occurrence Multi-words Towards Handling Very Large Spectrum of Text Corpora Sizes*

No Author Given

No Institute Given

March 2025

1 Introduction

This is a guide to reproduce the algorithms and the results related to the content of the paper titled *A Scalable Model for Frequency Distribution of Low Occurrence Multi-words Towards Handling Very Large Spectrum of Text Corpora Sizes*. File ArchivedNgrams.zip must be uncompressed and all files in it must be placed in the same folder. These are the files used to generate the results (except figures) reported in the paper:

Def_ValidationGHofC_Phase.py
Def_TestingGHofC_Phase.py
Def_ConstantsAndGlobalVars.py
Def_DistByLangNandK.py
Def_KThresolds.py
Def_TestingCorporaSizes.py
Def_ValidationCorporaSizes.py
Def_Vocabulary.py
Def_monotony.py
BaseLineModel.py
BaseLineModelDistKBD.py
BaseLineModelTrainingCorpora.py
BaseLineModelVocabulary.py
Def_smooth_spline_results.pkl
Def_smooth_spline_results_const.pkl

And the following are the files related to the generation of the figures in the paper:

GerFig1k1.txt, **powerlawVD1_k1.txt**, **powerlawVD1_k2.txt**, **strait_k1.txt**, **strait_k2.txt**, **GerFigDk.txt**, **EmmD1_2grams.txt**, **EmmD1_3grams.txt**, **EmmD2_2grams.txt**, **EmmD2_3grams.txt**, **PredD1_2grams.txt**, **PredD1_3grams.txt**, **PredD2_2grams.txt**, **PredD2_3grams.txt**, **GerFigWkbyC.txt**, **EmpWk11Gw.txt**,

EmpWk172Gw.txt, **EmpWk31Gw.txt**, **EmpWk366Mw.txt**, **EmpWk373Gw.txt**, **EmpWk82Gw.txt**, **PredWk11Gw.txt**, **PredWk172Gw.txt**, **PredWk31Gw.txt**, **PredWk366Mw.txt**, **PredWk373Gw.txt**, **PredWk82Gw.txt**.

Next sections explain how to reproduce the results of the paper.

2 Reproducing the Results of the Proposed Model

The results in the paper can be reproduced alternatively by:

a) Running the learning phase first, and then running the test phase.

b) Running the test phase based on the learning phase, which is already available.

Regarding *a)*, if we want to train the model for a language and an n-grams size, we must use python function **FindParametersCrossVal(Lang, NgramSize, Vi, Vf, Vs, InpercG, EnpercG, NstpG, InpercH, EnpercH, NstpH)** available in the file **Def_ValidationGHofC_Phase.py**. The meaning of the parameters are: the language; the n-gram size; the initial, the final and the step values for vocabulary size searching; the initial, the final and the number of steps values for parameter g_k (relative values around estimated points performed by the algorithm as explained in the paper); the initial, the final and the number of steps values for parameter h_k (relative values around estimated points performed by the algorithm as explained in the paper). These parameters are respectively for Lang, NgramSize, Vi, Vf, Vs, InpercG, EnpercG, NstpG, InpercH, EnpercH and NstpH. As an example, we can run

```
FindParametersCrossVal('en',2,1.985e10,2.00e10,1e8,0.4,1.7,400,0.4,1.7,200)
```

to train the model for English 2-grams. Then we will see:

```
It may take some minutes
k_thresold for training corpora: 17
Processing for the hypothetical vocabulary size: 19850000000
Processing for the hypothetical vocabulary size: 19950000000
Best Vocabulary Size: 19950000000
```

The model can also be trained for German ('de') (since there are empirical n-gram counts available also for German in file **Def_DistByLangNandK.py**). Other n-gram sizes (1,3,4,5,6) are also considered.

This function performs: the estimation of the vocabulary size and saves it in file **Def_Vocabulary.py**; calculates the k-threshold value – file **Def_monotony.py** is used for that – and saves it in file **Def_KThresolds.py**; saves the g_k and h_k values resulting from *splines* and saves them in file **Def_smooth_spline_results.pkl**. After the training phase, results can be obtained as explained next.

Regarding *b)* or for the case when training has already been done, results can be obtained by using some functions of file **Def_TestingGHofC_Phase.py**.

2.1 D_k Predictions for each *Corpus* Size

In order to obtain the $D(k, C; L, n)$ values (the number of distinct n-grams with frequency greater or equal to k, for a *corpus* of size C, for language L and n-gram size n), we must run **AvgDkRelErrorForEachTestCorpusWithScale(L,n)**. For English 2-grams, for example, it will be *AvgDkRelErrorForEachTestCorpusWithScale('en',2)*. The results will be:

```
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 366397190 : 0.008778096245756954 0.012501532643591398
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 11344756226 : 0.005131186374030359 0.007755717180264833
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 31487751849 : 0.0013902488480408 73 0.0016880610900871554
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 82718285912 : 0.003753852597988539 0.004208358650843619
Global Averages for Relative Errors, for C: 0.004763346016454182
```

The *corpora* sizes shown belong to the test set. Other n-gram sizes (1,3,4,5,6) can also be used, as well as German ('de').

2.2 D_k Predictions for each k Value

To obtain the $D(k, C; L, n)$ values for each k value, we must run (example for German 1-grams) **EachDkRelErrorTroughTestCorporaWithScale('de',1)**. The result will be:

```
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 1 : 0.008490530174184 53 0.012186506428339 25
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 2 : 0.006720067208627 5585 0.009679657957020624
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 3 : 0.007771722277930806 0.012673282797699581
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 4 : 0.009088328578489901 0.015655086012544086
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 5 : 0.010384346674730975 0.018366500908390555
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 6 : 0.010539299564900937 0.019420349157567 24
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 7 : 0.009905875239372726 0.018642073728824215
```

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 8 : 0.010362760323039290 0.019552127288343102

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 9 : 0.01015909784412201 0.0190171907397087

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 10 : 0.010158350377593947 0.01915349820125297

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 11 : 0.00984579931560585 0.017586457472024166

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 12 : 0.009340617522387946 0.016450594745856965

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 13 : 0.008641381239376428 0.015496191975991542

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 14 : 0.008342976111642999 0.015271078536159057

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 15 : 0.007495007791075804 0.013901153749269816

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 16 : 0.007320675764995028 0.013118185838602345

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 32 : 0.006133040842653309 0.011437557708962582

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 64 : 0.002435957309486738 0.002946845752580633

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 128 : 0.0008157786530782570 0.0012061565512207256

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 256 : 0.0012151504942092381 0.0014729533324368922

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 512 : 0.004117180032897926 0.005526034927606230

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 1024 : 0.006165708488898545 0.008875552862705962

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 2048 : 0.0029237630907731653 0.004250603817119361

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 4096 : 0.0028995218559839090 0.0031710731719632405

Global Averages for Relative Errors for Dk: 0.007136372365669076

2.3 W_k Predictions for each *Corpus* Size

To obtain the $W(k, C; L, n)$ values (the number of distinct n-grams with frequency equal to k, for a *corpus* of size C, for language L and n-gram size n), we must run (example for English 3-grams) **AvgWkRelErrorForEachTestCorpusWithScale('en',3)**. Result will be:

Prediction of Relative Errors and Mean Square Root of wk Relative Errors for C = 366397189 : 0.018642560759557277 0.023954128241150348

Prediction of Relative Errors and Mean Square Root of wk Relative Errors for C = 11344756219 : 0.011709805185349254 0.01648502262704967

Prediction of Relative Errors and Mean Square Root of wk Relative Errors for C = 31487751831 : 0.0042812559756207685 0.006066230839741724

Prediction of Relative Errors and Mean Square Root of wk Relative Errors for C = 82718285866 : 0.0034558206248180624 0.005236572183512354

Global Averages for Relative Errors, for C: 0.00952236063633634

2.4 W_k Predictions for each k Value

To obtain the $W(k, C; L, n)$ values for each k value, we must run (example for German 4-grams) **EachWkRelErrorTroughTestCorporaWithScale('de',4)**. The result will be:

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 1 : 0.006245231763732335 0.009674409157842576

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 2 : 0.008660560169364485 0.015195507344587841

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 3 : 0.014151072049539562 0.02617537350252586

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 4 : 0.00797595253373079 0.009940075914550626

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 5 : 0.003644634169138551 0.0062387260556056

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 6 : 0.03046110800193105 0.05306780442537013

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 7 : 0.0047830080276595 0.0071817324271585365

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 8 : 0.009510117730532548 0.015004339364116112

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 9 : 0.00506753558088398 0.006639749181006347

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 10 : 0.0083602061303187 0.012146890712848066

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 11 : 0.05336989963602594 0.10281428610760057

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 12 : 0.024837679346496666 0.04645613836591567

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 13 : 0.03768685628978186 0.06910938113726507

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 14 : 0.04719759931729035 0.08955720774117897

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 15 : 0.019518677205597842 0.027517275731672747

Prediction of Relative Errors and Mean Square Root of Relative Errors for Wk, k = 16 : 0.031106685796940845 0.06123668509236676

```
    Prediction of Relative Errors and Mean Square Root of Relative Errors for
Wk, k = 17 : 0.011538907135197422 0.016462212374322608
    Global Averages for Relative Errors, for Wk: 0.01906563122848014
```

3 Reproducing the Results of the Baseline Model

To reproduce the results of the Baseline model, some functions of the file **BaseLineModel.py** must be used.

3.1 D_k Predictions for each *Corpus* Size

To reproduce these results, we must run (example for English 2-grams)**AntModTestDkActualC('en', 2)**. The results will be:

```
    Dk average and square Errors for c,Lang, Ngram: 366397190 en 2 : 0.3479429689129384
0.35150103822720175
    Dk average and square Errors for c,Lang, Ngram: 11344756226 en 2 : 0.1845444617935846
0.1882913860994461
    Dk average and square Errors for c,Lang, Ngram: 31487751849 en 2 : 0.12648210955272876
0.1298042564802903
    Dk average and square Errors for c,Lang, Ngram: 82718285912 en 2 : 0.06189720790184233
0.08666983670089452
    Global Error : 0.18021668704027355
```

3.2 D_k Predictions for each k Value

These results will be given by function **AntModTestDkActualKbyK('en', 1)** (example for English 1-grams):

```
    Dk average and square Errors for k,Lang, Ngram: 1 en 1 : 0.8743386248218455
0.9280331731987299
    Dk average and square Errors for k,Lang, Ngram: 2 en 1 : 0.7197735260330508
0.7355192031933593
    Dk average and square Errors for k,Lang, Ngram: 3 en 1 : 0.677761189943357
0.6922814990487511
    Dk average and square Errors for k,Lang, Ngram: 4 en 1 : 0.6405244316106208
0.6560912454585672
    Dk average and square Errors for k,Lang, Ngram: 5 en 1 : 0.6126767963340082
0.628579381322395
    Dk average and square Errors for k,Lang, Ngram: 6 en 1 : 0.5789157863016462
0.5943161024109519
    Dk average and square Errors for k,Lang, Ngram: 7 en 1 : 0.5660245284792901
0.5812936360351759
    Dk average and square Errors for k,Lang, Ngram: 8 en 1 : 0.541280214691845
0.5561677491848851
```

```
Dk average and square Errors for k,Lang, Ngram: 9 en 1 : 0.5298732155588947 0.5438293138686668
Dk average and square Errors for k,Lang, Ngram: 10 en 1 : 0.5214961551405822 0.5357427230197834
Dk average and square Errors for k,Lang, Ngram: 11 en 1 : 0.5135993440667627 0.5278455160189464
Dk average and square Errors for k,Lang, Ngram: 12 en 1 : 0.5012997893199591 0.5154477738561042
Dk average and square Errors for k,Lang, Ngram: 13 en 1 : 0.4946363988982175 0.5087963524025215
Dk average and square Errors for k,Lang, Ngram: 14 en 1 : 0.4830666633430951 0.49712056229538365
Dk average and square Errors for k,Lang, Ngram: 15 en 1 : 0.47537336595905078 0.4895635307797699
Dk average and square Errors for k,Lang, Ngram: 16 en 1 : 0.4660570506715273 0.48021497098449234
Global Error 0.574793567573336
```

3.3 W_k Predictions for each *Corpus* Size

The results for these predictions are obtained by function **AntModTestWkActualC('en', 4)** (example for English 4-grams). We will see:

```
Wk average and square Errors for c,Lang, Ngram: 366397188 en 4 : 0.4545753806415851 0.4581471864432782
Wk average and square Errors for c,Lang, Ngram: 11344756212 en 4 : 0.4045925221368847 0.40777075747775654
Wk average and square Errors for c,Lang, Ngram: 31487751813 en 4 : 0.3965766410106921 0.40271421459657564
Wk average and square Errors for c,Lang, Ngram: 82718285820 en 4 : 0.37375428196749866 0.38146687989443634
Global Error : 0.4073747064391651
```

3.4 W_k Predictions for each k Value

For this case, function **AntModTestWkActualKbyK('en', 3)** must be run (example for English 3-grams):

```
Wk average and square Errors for k,Lang, Ngram: 1 en 3 : 0.1382600224269316 0.1902472636337762
Wk average and square Errors for k,Lang, Ngram: 2 en 3 : 0.34537791251963024 0.3508103865959475
Wk average and square Errors for k,Lang, Ngram: 3 en 3 : 0.3418870326112877 0.3475676970818502
Wk average and square Errors for k,Lang, Ngram: 4 en 3 : 0.35771802276845643 0.3641902048361738
Wk average and square Errors for k,Lang, Ngram: 5 en 3 : 0.3462752998748535 0.3572764485904226
Wk average and square Errors for k,Lang, Ngram: 6 en 3 : 0.37926604292987576 0.39026615743448617
Wk average and square Errors for k,Lang, Ngram: 7 en 3 : 0.3462688999028242 0.3597628116842053
Wk average and square Errors for k,Lang, Ngram: 8 en 3 : 0.37740542837408864 0.3908476225002481
Wk average and square Errors for k,Lang, Ngram: 9 en 3 : 0.3531057187931252 0.36728618720761624
Wk average and square Errors for k,Lang, Ngram: 10 en 3 : 0.3582794520047582 0.37359439938138944
Wk average and square Errors for k,Lang, Ngram: 11 en 3 : 0.33474179568939627 0.3484183770127716
Wk average and square Errors for k,Lang, Ngram: 12 en 3 : 0.35322647409971497 0.364475569232437
Wk average and square Errors for k,Lang, Ngram: 13 en 3 : 0.33295945209532085 0.34348130203108995
Wk average and square Errors for k,Lang, Ngram: 14 en 3 : 0.3362720252764011 0.3507500373652505
Wk average and square Errors for k,Lang, Ngram: 15 en 3 : 0.32715874905392506 0.3388055126595562
Global Error 0.33521348856137273
```

4 Reproducing the Results for the Model Using Constant Parameters and Cross-Validation

As for the case of the proposed model in Section 2, results can be obtained by using the training phase, which is already available, or by training this model before the test phase.

For training the model, function **FindParametersCrossVal('en', 2, 1.985e10, 2.00e10, 1e8, 0.4, 1.7, 400, 0.4, 1.7, 200, "ConstParms")** (example for English 2-grams) from file **Def_ValidationGHofC_Phase.py** must be run. This is the same function as the one used in Section 2, so the meaning of the parameters are the same. Although, an additional parameter (*"ConstParms"*) indicates a different model. The following results will be seen:

```
It may take some minutes
k_thresold for training corpora: 17
Processing for the hypothetical vocabulary size: 19850000000
Processing for the hypothetical vocabulary size: 19950000000
Best Vocabulary Size: 19950000000
```

4.1 D_k Predictions for each *Corpus* Size

To obtain results for these predictions, function **AvgDkRelErrorForEachTestCorpusWithScale('en', 6, "ConstParms")** (example for English 6-grams) from file **Def_TestingGHofC_Phase.py** is used. The results will be:

Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 366397186 : 0.5470557736344195 0.7215757455003925
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 11344756198 : 0.06257949916140662 0.07249043107739996
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 31487751777 : 0.04731543884581135 0.0581867394836642
Prediction of Relative Errors and Mean Square Root of Dk Relative Errors for C = 82718285728 : 0.013735208140885436 0.018957521096334837
Global Averages for Relative Errors, for C: 0.16767147994563072

4.2 D_k Predictions for each k Value

For this case, function **EachDkRelErrorTroughTestCorporaWithScale('de', 5, "ConstParms")** (example for German 5-grams) must be called. Results will be:

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 1 : 0.06319805793682381 0.10405157721696671
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 2 : 0.005648711233361225 0.009821252460839189
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 3 : 0.03674619459448325 0.06447605713897969
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 4 : 0.06282833195922559 0.1087022226471926
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 5 : 0.0699172847664049 0.12178375077375486
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 6 : 0.0780673370477064 0.13690189952688064
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 7 : 0.0801316005338459 0.14159154554663253
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 8 : 0.08539460377031968 0.15099329265283332
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 9 : 0.08790645374002296 0.15661610612417642
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 10 : 0.09226138990755157 0.1645535068245945
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 11 : 0.09332739273300093 0.16744458121640882
Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 12 : 0.0911111564703024 0.16294845156391152

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 13 : 0.09115990143855714 0.16436931356674866

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 14 : 0.089102233819684 0.16172553411509744

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 15 : 0.08838401794705207 0.15973252999412205

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 16 : 0.0854504108395549 0.1564568380716739

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 32 : 0.06808877653474203 0.13216717167706035

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 64 : 0.10139610651165959 0.18827009953977752

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 128 : 0.12182394304210789 0.19675882812783702

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 256 : 0.12857949130845694 0.19010648621965315

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 512 : 0.2390661680930702 0.4271998592496243

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 1024 : 0.160967776664164420 0.2948498827365511

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 2048 : 0.4429876756216375 0.8774280399110012

Prediction of Relative Errors and Mean Square Root of Relative Errors for Dk, k >= 4096 : 0.7152260237005563 1.3498023552781868

Global Averages for Relative Errors for Dk: 0.13244879334226214

5 Reproducing the Figures

Figure 1 of the paper, can be generated by running *load "GerFig1k1.txt"* in *gnuplot* context. File *GerFig1k1.txt* uses files: *powerlawVD1_k1.txt*, *powerlawVD1_k2.txt*, *strait_k1.txt* and *strait_k2.txt* for that.

Figure 2a can be generated by running *load "GerFigDk.txt"* in *gnuplot* context. File *GerFigDk.txt* uses files: *EmmD1_2grams.txt*, *EmmD1_3grams.txt*, *EmmD2_2grams.txt*, *EmmD2_3grams.txt*, *PredD1_2grams.txt*, *PredD1_3grams.txt*, *PredD2_2grams.txt*, *PredD2_3grams.txt* for that.

Figure 2b can be obtained by running *load "GerFigWkbyC.txt"* in *gnuplot* context. File *GerFigWkbyC.txt* uses files: *GerFigWkbyC.txt*, *EmpWk11Gw.txt*, *EmpWk172Gw.txt*, *EmpWk31Gw.txt*, *EmpWk366Mw.txt*, *EmpWk373Gw.txt*, *EmpWk82Gw.txt*, *PredWk11Gw.txt*, *PredWk172Gw.txt*, *PredWk31Gw.txt*, *PredWk366Mw.txt*, *PredWk373Gw.txt*, *PredWk82Gw.txt*.