Universidade de Aveiro

Departamento de Electrónica, Telecomunicações e Informática

Algorithmic Information Theory (2021/22)

Lab work n.º 2

Grupo n.º 10

Fernando Lopes n.º 106358

Maria João Sousa n.º 109488

João Carvalho n.º 89059

Índice

Introdução

Neste trabalho é abordado como problema principal a “similaridade” entre textos, um ficheiro target e ficheiros de referência. Mais especificamente, a “similaridade” diz respeito às línguas nas quais os ficheiros estão escritos.

Para a resolução deste problema foi usado um algoritmo de compressão de informação com base no algoritmo desenvolvido no primeiro projeto(*finite-context model - FCM*).

A ideia principal é usar modelos descritivos dos textos referência para calcular uma estimativa do número de bits necessários para a compressão do ficheiro *target*, e consequentemente medir a “similaridade” entre eles.

Foram desenvolvidos dois programas principais - ***FindLang*** e ***LocateLang*** - baseados num terceiro programa - ***Lang.*** Além disso foi ainda implementado um desafio bónus que explora as funcionalidades desenvolvidas. Todos os passos de implementação, ideias de desenvolvimento e análise dos resultados serão descritos ao longo do relatório.

Finite-context model

Os modelos *FCM* são utilizados para comparar o texto de referência com o texto que o utilizador pretende analisar. Estes modelos são criados caso não existam e são utilizados da memória caso o programa já tenha sido executado previamente utilizando os mesmos ficheiros de referência, o mesmo valor de K e o mesmo valor de alpha. Ao guardar os dados gerados pelo *FCM*, o tempo de execução do programa fica significativamente menor.

K	Tempo de execução de FCM (em segundos)	Tempo de ir buscar informação do FCM à memória (em segundos)
1	45.305	0.051
2	54.865	0.187
3	58.814	0.711
4	69.371	2.192
5	87.674	5.946
6	113.682	11.911

7	145.599	18.135
8	174.939	27.202
9	207.894	36.991
10	250.037	50.087

Tabela 1 - Comparação de tempos de execução. O exemplo considerado foi o Great Gatsby. Alpha = 0.1

A partir do modelo de *FCM* gerado para cada um dos ficheiros de referência, é possível comparar com os exemplos (ficheiros target) dados pelo utilizador.

Calculando uma estimativa do número de bits necessários para comprimir o exemplo dado pelo utilizador, através dos *FCMs* dos ficheiros de referência, é possível descobrir em que língua é que o exemplo foi escrito. Se o ficheiro dado pelo utilizador for escrito em Português, as sequências de caracteres serão idênticas às sequências do ficheiro de referência de língua portuguesa, fazendo com que o *FCM* da referência do Português seja o modelo que necessita de menos bits para comprimir o ficheiro target, sendo assim possível detectar em que língua foi escrito o texto exemplo do utilizador, se este constar nas referências.

Deteção da língua

Como abordagem para a resolução do problema da “similaridade” foram usados vinte e um textos como referência, isto é, vinte e um textos escritos em línguas diferentes, representativos das mesmas.

Foram desenvolvidos três programas diferentes - ***Lang***, ***Findlang*** e ***Locatelang***.

1. *Lang*

O ***Lang*** é o programa base para os restantes. Dando um ficheiro de referência(*ref*) e um ficheiro *target*(t), o programa retorna uma estimativa do número de bits necessários para comprimir *t* com base no *ref.* A principal função implementada no programa percorre cada símbolo e seu contexto e, caso esse contexto e símbolo estejam presentes no modelo(*FCM*) de *ref*, utiliza as probabilidades desse modelo para estimar o número de bits, caso contrário calcula-se a nova probabilidade e consequentemente o número de bits a partir dela. Por fim, apenas calcula-se a soma dos bits de cada símbolo.

2. *FindLang*

No ***Findlang*** pretende-se saber apenas a que língua o texto exemplo que o utilizador introduziu se assemelha mais. O programa faz a comparação do *FCM* como descrito acima para cada uma das referências e guarda esses valores num dicionário com os nomes das referências como key e a soma dos bits calculados para cada símbolo - recorrendo ao programa *Lang* - como value. Esse dicionário é então ordenado de forma a encontrar qual a referência cuja soma de bits é menor e o nome da referência (língua) é retornada ao utilizador.

Resultados e análise - Findlang

Para demonstrar a funcionalidade que descobre em que língua está escrito um ficheiro (*Findlang*), foram utilizados dois ficheiros de texto. Para cada um dos ficheiros fez-se variar o K entre 3 e 9 e o alpha entre 0.1 e 1.
Na tabela estão apenas as 3 referências que obtiveram os melhores resultados na compressão para cada exemplo. A língua que necessita de menos Kbs para comprimir o ficheiro será a língua do ficheiro.

Examples:size(Kb)	Language	K	Alpha	Language References	Size(Kb)
Gatsby.txt: 299Kb	English	3	0.1	English	133.31
				Dutch	214.06
				Danish	220.22
			1	English	127.85
				Dutch	147.18
				Danish	202.55
		9	0.1	English	216.06
				Swedish	253.11
				Italian	253.79
			1	English	214.68
				Swedish	253.07
				Italian	253.68
			0.1	Portuguese	1.14
				Spanish	1.77

Test.txt: 3.21Kb	Portuguese	3		Italian	2.18
			1	Portuguese	1.11
				Spanish	1.65
				Italian	2.01
		9	0.1	Portuguese	1.83
				Spanish	2.54
				Italian	2.71
			1	Portuguese	1.81
				Spanish	2.53
				Italian	2.71

Tabela 2 - Demonstração do *Findlang* para dois ficheiros de texto variando o K e o alpha

Para todos exemplos demonstrados o *Findlang* detectou corretamente as línguas em que os ficheiros de texto estão escritos.
O K = 3 gerou melhores resultados que o K = 9, necessitando de menos bits para comprimir os ficheiros. Em relação ao alpha, o alpha = 1 teve ligeiramente melhores resultados que o alpha = 0.1 na compressão, mas na deteção da língua do ficheiro os resultados foram semelhantes.

3. *LocateLang*

No ***Locatelang***, assim como no *Findlang*, o programa faz a comparação do *FCM* como descrito acima para cada uma das referências e guarda esses valores num dicionário com os nomes das referências como key porém, o value é a lista de todos os bits calculados para cada símbolo, recorrendo mais uma vez ao programa *Lang*. Como os valores dos bits que são recebidos após a comparação com o modelo de *FCM* não são contínuos, é feita uma suavização dos mesmos. Isto permite que a previsão da linguagem tenha menos falhas e que a deteção seja mais acertada. A suavização consiste em fazer a média dos últimos *len(bits) / 50* valores ou da totalidade do texto caso este seja de dimensões reduzidas. Após este processo, é calculada a média da entropia para cada uma das referências e é escolhida a que tem menor valor, ou seja, a que tem maior probabilidade de corresponder com a linguagem de referência. O menor valor da média da entropia é então multiplicado por 1.15 e esse valor passa a ser o *threshold*. Os valores abaixo do *threshold* são considerados como certos, ou seja, a linguagem do exemplo passado pelo utilizador corresponde à linguagem da referência e as posições desses valores são mostradas ao utilizador via terminal. Os valores acima do *threshold* são considerados errados e por isso não são mostrados. É ainda considerado um tratamento de outliers aquando da criação da lista de resultados - caso os valores que não correspondam à referência sejam de comprimento menor a 40, não é considerada uma mudança de linguagem. Da mesma forma, se o comprimento de valores considerados certos for menor que 10, estes não são considerados. Mesmo com todo este tratamento de dados, há casos em que línguas que não estão presentes no ficheiro target podem aparecer no resultado final, uma vez que

apresentam um o alfabeto e contextos bastante semelhantes a uma determinada língua que, de facto, está contida no texto. Contudo, foi notado que, no resultado final, o intervalo de caracteres das línguas não presentes estava contido no intervalo da língua semelhante presente no texto, deste modo, e para resolver esta questão, for implementado um ciclo que verificava a presença de subsets de intervalos e, se houver casos, não se leva em consideração essa língua(subset) no resultado final.

Resultados e análise - *LocateLang* - K = 3, alpha = 0.1

- *Gatsby.txt* (Inglês)

Langs in the target text:
English starts at char 0 and ends at char 289885
Time: 469 seconds

Representação gráfica das línguas encontradas no texto:

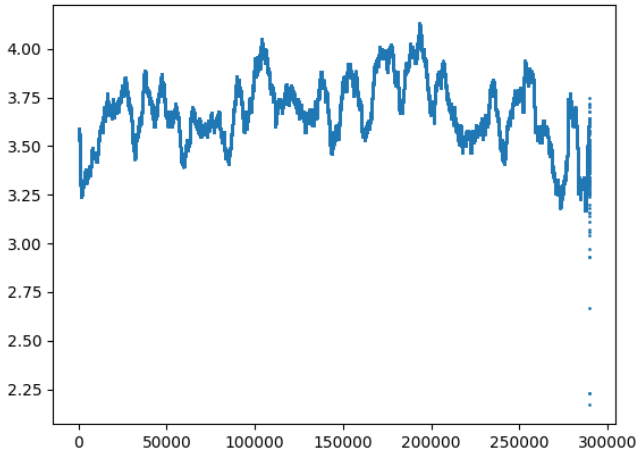

Imagem 1 - Gráfico com valores inferiores ao threshold para a lingua Inglesa correspondente ao ficheiro “Gatsby.txt”

- *deu_en.txt* (Alemão - Inglês)

Langs in the target text:
German starts at char 0 and ends at char 5065
English starts at char 5000 and ends at char 14140

Time: 2.47 seconds

Representação gráfica das línguas encontradas no texto:

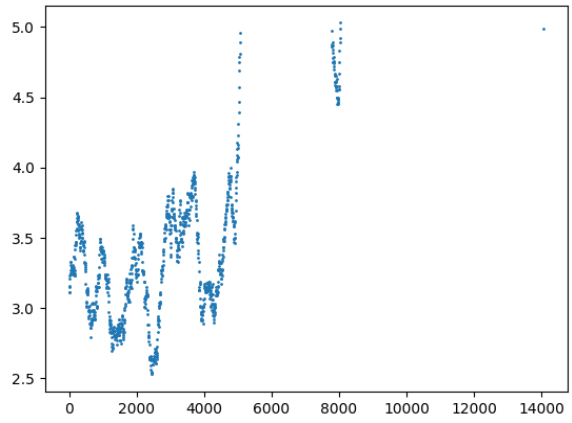

Imagem 2 - Gráfico com valores inferiores ao threshold para a lingua Alemâ correspondente ao ficheiro “deu_en.txt”

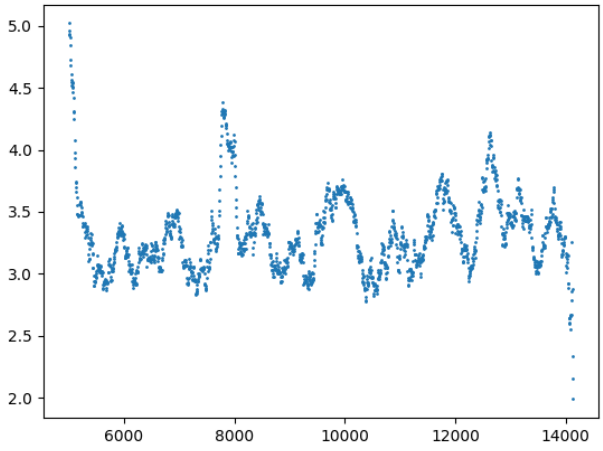

Imagem 3 - Gráfico com valores inferiores ao threshold para a lingua Inglesa correspondente ao ficheiro “deu_en.txt”

- *en_es.txt* (Inglês - Espanhol)

Langs in the target text:
English starts at char 0 and ends at char 30585
Spanish starts at char 30705 and ends at char 38350

Time: 12.27 seconds

Representação gráfica das línguas encontradas no texto:

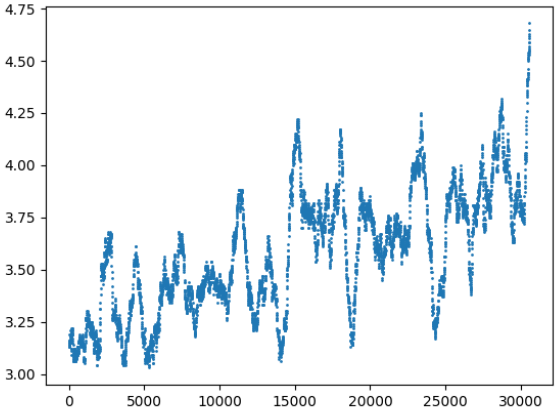

Imagem 4 - Gráfico com valores inferiores ao threshold para a lingua Inglesa correspondente ao ficheiro “en_es.txt”

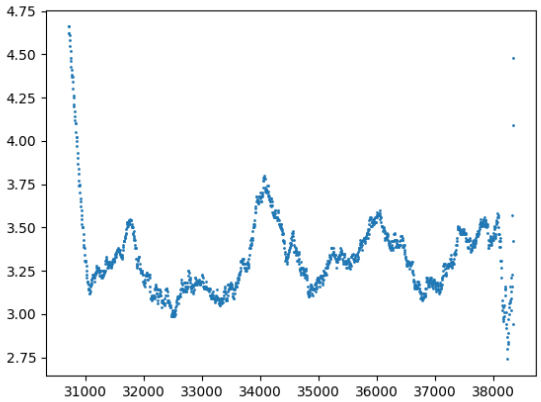

Imagem 5 - Gráfico com valores inferiores ao threshold para a lingua Espanhola correspondente ao ficheiro “en_es.txt”

- *mix_PT_DEU_EN.txt* (Alemão - Português - Alemão - Inglês)

Langs in the target text:
German starts at char 0 and ends at char 3645
German starts at char 7935 and ends at char 11455
English starts at char 10930 and ends at char 15575
Portuguese starts at char 3485 and ends at char 8055

Time: 3.61 seconds

Para se ter uma noção da precisão destes intervalos, abaixo encontram-se os intervalos **exatos** de mudanças de língua no texto target “*mix_PT_DEU_EN.txt*”.

German starts at char 0 and ends at char 3708
German starts at char 8158 and ends at char 11180

English starts at char 11181 and ends at char 15588
Portuguese starts at char 3709 and ends at char 8157

Abaixo encontram-se os resultados do texto target "*mix_PT_DEU_EN.txt*" com **K = 7** e **alpha = 0.1** de maneira a conseguirmos fazer uma análise com um valor diferente de k.

German starts at char 0 and ends at char 3665
German starts at char 7860 and ends at char 11380
English starts at char 10915 and ends at char 15570
Portuguese starts at char 3420 and ends at char 8125

Time: 32.02431893348694

Pode-se analisar que os valores não são muito distantes aos do k com valor 3 e, comparando aos valores exatos, nota-se uma ligeira **melhora na precisão dos resultados**.

Representação gráfica das línguas encontradas no texto:

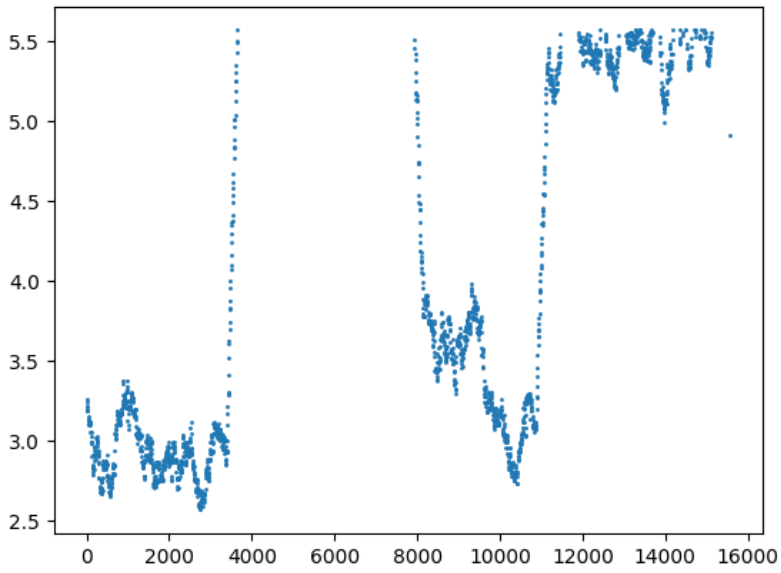

Imagem 6 - Gráfico com valores inferiores ao threshold para a lingua Alemã correspondente ao ficheiro "mix_PT_DEU_EN.txt"

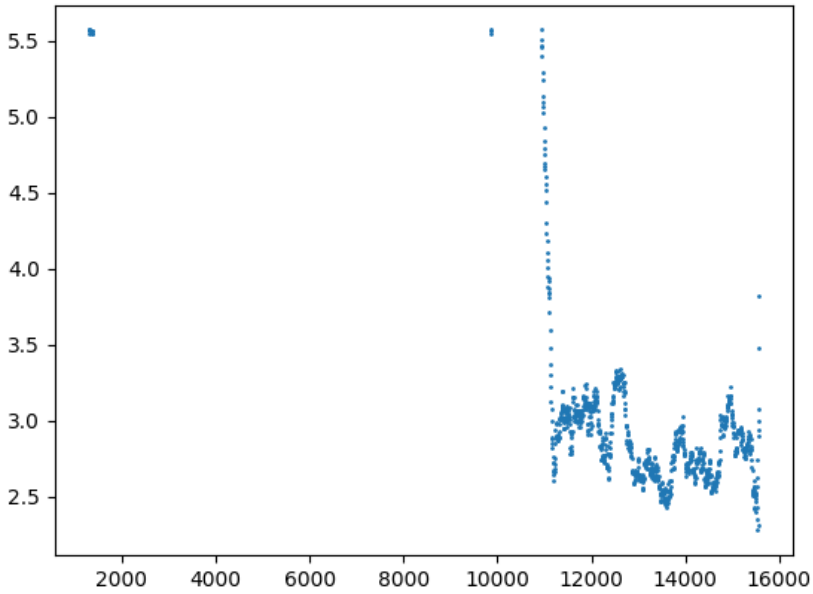

Imagem 7 - Gráfico com valores inferiores ao threshold para a lingua Inglesa correspondente ao ficheiro “mix_PT_DEU_EN.txt”

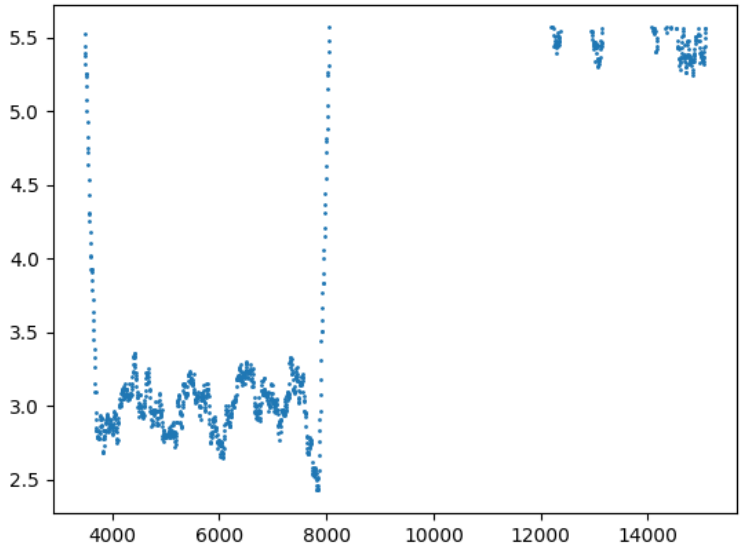

Imagem 8 - Gráfico com valores inferiores ao threshold para a lingua Portuguesa correspondente ao ficheiro “mix_PT_DEU_EN.txt”

Desafio bónus

O desafio bónus considera 2 valores distintos de K, 1 ficheiro de referência e 1 ficheiro de exemplo. O processo de geração do modelo de FCM, comparação com a referência e o smoothing são feitos à semelhança do programa default, com a diferença que o processo é feito para 2 valores de K distintos, em vez de apenas 1. Com as listas de valores de entropia para cada caráter podemos comparar posição a posição para qual lista a posição tem menor valor. O menor valor representa um melhor desempenho e por isso é selecionado e é posto numa lista de resultados, juntamente com o número da lista de onde foi selecionado. Este processo consegue, de forma geral, comprimir mais os dados, uma vez que a soma do número de bits da lista de resultado é menor que a soma do número de bits da lista K1 e do que a soma do número de bits da lista K2.

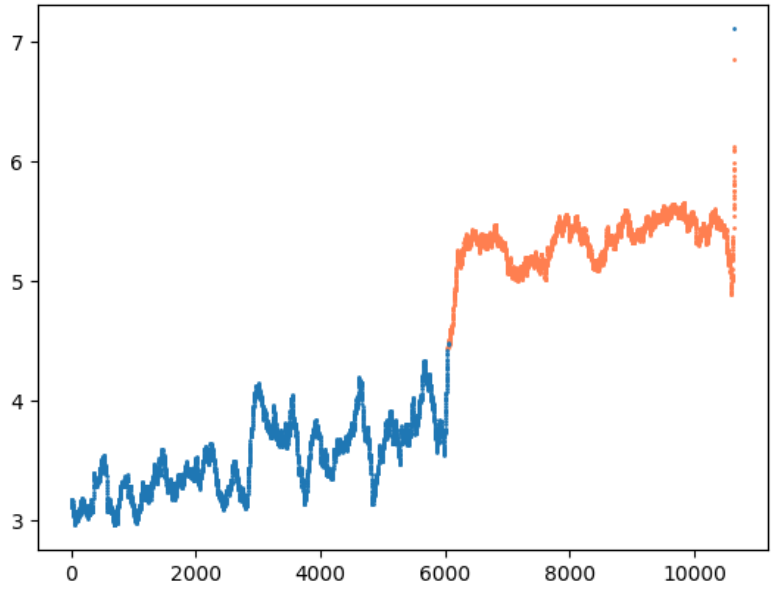

Gráfico 1 - Referência - eng_GB.latn.English.EP7.utf8; Exemplo - en_pt.txt (gatsby.txt + maias.txt); K = 2 e K = 4; Alpha = 0.1; A laranja estão representados os valores em que K = 2 teve um melhor desempenho, a azul estão representados os valores em que K = 4 teve um melhor desempenho.

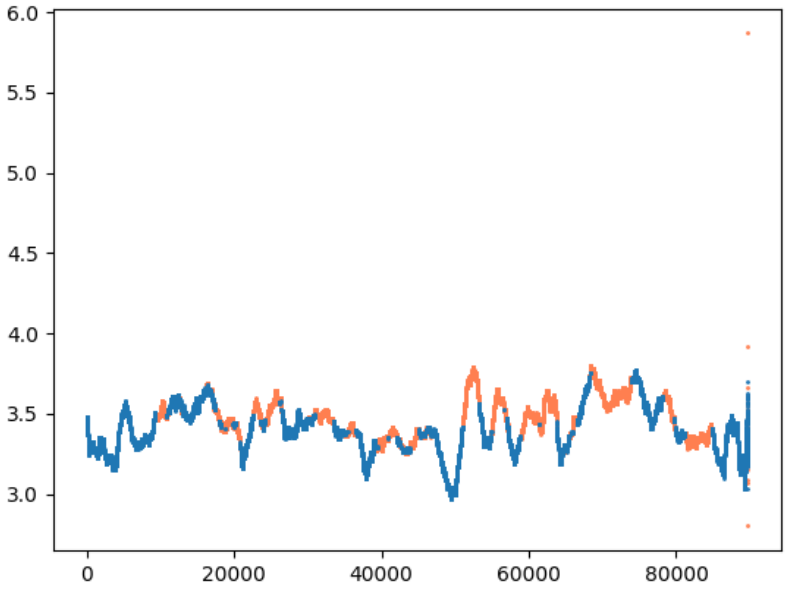

Gráfico 2 - Referência - eng_GB.latn.English.EP7.utf8; Exemplo - dorian.txt; K = 3 e K = 4; Alpha = 0.1; A laranja estão representados os valores em que K = 3 teve um melhor desempenho, a azul estão representados os valores em que K = 4 teve um melhor desempenho.

Percentagens de participação

Fernando Lopes n.º 106358 - 33%

Maria João Sousa n.º 109488 - 33%

João Carvalho n.º 89059 - 33%

Links utilizados

The Picture of Dorian Gray(dorian.txt) - https://www.gutenberg.org/ebooks/174
The Great Gatsby(gatsby.txt) - https://www.gutenberg.org/ebooks/64317
Dracula (dracula_es.txt) - http://bibliotecadigital.ilce.edu.mx/Colecciones/ObrasClasicas/_docs/Dracula_Stoker.pdf
Os maias(maias.txt) - http://bibliotecasicl.pt/Biblionet/Services/GetRepositoryFile.ashx?repository=105199_REPOSITORY-BDIGITAL&guid=a7d99695-1d3c-430e-8ea6-01e137b7f838
DEU.txt: https://lingua.com/german/reading/
EN.txt: https://lingua.com/english/reading/