Processamento de Linguagem Natural
Trabalho Prático 1 - Modelos de Linguagem

Aluno: Breno de Sousa Matos
Matrícula: 2017086007

1.Introdução

Este trabalho tem como objetivo avaliar diferentes modelos de linguagem neuronais.

2.Metodologia

Para este trabalho, foi obtido um corpus [1], produzidos vários modelos word2vec diferentes, fazendo variações dos tamanho do corpus e contexto, além de variações entre os algoritmos skip-gram e CBOW. Apesar de ter sido disponibilizado um código em C, foi utilizada a biblioteca *Gensim* [2] para python3.

Para realizar a avaliação de cada modelo criado, foi utilizado o arquivo *questions-words.txt*, fornecido juntamente com a implementação de word2vec em C fornecida em [3]. Em seguida, são fornecidas as três primeiras palavras de cada linha do arquivo ao modelo gerado. O resultado correto para cada linha de teste é a quarta palavra da mesma linha. Foi então calculado o erro com base na diferença entre as distâncias da palavra retornada no topo do ranking gerado e da palavra correta.

Foram elaborados 30 modelos diferentes, utilizando os seguintes parâmetros para a função *gensim.model.Word2Vec*:

- corpus: Dado o corpus fornecido, este foi variado entre 50%, 75% e 100% de sua composição original.
- window: Tamanho do contexto utilizado para treinar o modelo, variado entre 2,4,6,8 e 10 palavras.
- min_count: Fixado em 1, para considerar palavras com pelo menos uma ocorrência, a fim de considerar o maior número possível de palavras.
- iter: Para todos os modelos gerados, foram realizadas 5 iterações de treino.
- sg: Parâmetro para definir se o algoritmo usado para treinamento do modelo será skip-gram (sg = 1) ou CBOW (sg = 0).

Além do erro calculado, foi também utilizada a função *accuracy,* também da biblioteca *Gensim*, a fim de calcular a porcentagem total de acertos do modelo ao tentar predizer qual a quarta palavra dadas outras três, utilizando o arquivo *questions-words.txt*.

Resultados

3.1 Tabela 1

Os resultados obtidos para cada modelo podem ser observados na **Tabela 1** a seguir:

Modelo	Corpus (%)	Contexto	Skip-gram	Erro	Acertos (%)
corpus50-2-0	50	2	0	0.058121	15.6
corpus50-2-1	50	2	1	0.040061	17.7
corpus50-4-0	50	4	0	0.062288	16.9
corpus50-4-1	50	4	1	0.040280	21.3
corpus50-6-0	50	6	0	0.064735	17.4
corpus50-6-1	50	6	1	0.041495	22.7
corpus50-8-0	50	8	0	0.066328	17.9
corpus50-8-1	50	8	1	0.041472	23.5
corpus50-10-0	50	10	0	0.066056	19
corpus50-10-1	50	10	1	0.041294	25.9
corpus75-2-0	75	2	0	0.055993	21.5
corpus75-2-1	75	2	1	0.039082	24.8
corpus75-4-0	75	4	0	0.058878	24.6
corpus75-4-1	75	4	1	0.038247	29.2
corpus75-6-0	75	6	0	0.059933	25.7
corpus75-6-1	75	6	1	0.038523	32.2
corpus75-8-0	75	8	0	0.060471	27
corpus75-8-1	75	8	1	0.038245	34.1
corpus75-10-0	75	10	0	0.060950	27
corpus75-10-1	75	10	1	0.038653	34.2
corpus100-2-0	100	2	0	0.054539	25.8
corpus100-2-1	100	2	1	0.037189	29
corpus100-4-0	100	4	0	0.055352	30.1
corpus100-4-1	100	4	1	0.035785	34.3
corpus100-6-0	100	6	0	0.055352	32.2
corpus100-6-1	100	6	1	0.035621	37.8
corpus100-8-0	100	8	0	0.056056	33.1
corpus100-8-1	100	8	1	0.035837	39.9
corpus100-10-0	100	10	0	0.056413	33.7
corpus100-10-1	100	10	1	0.035390	41.3

Abaixo, o significado de cada campo da tabela:

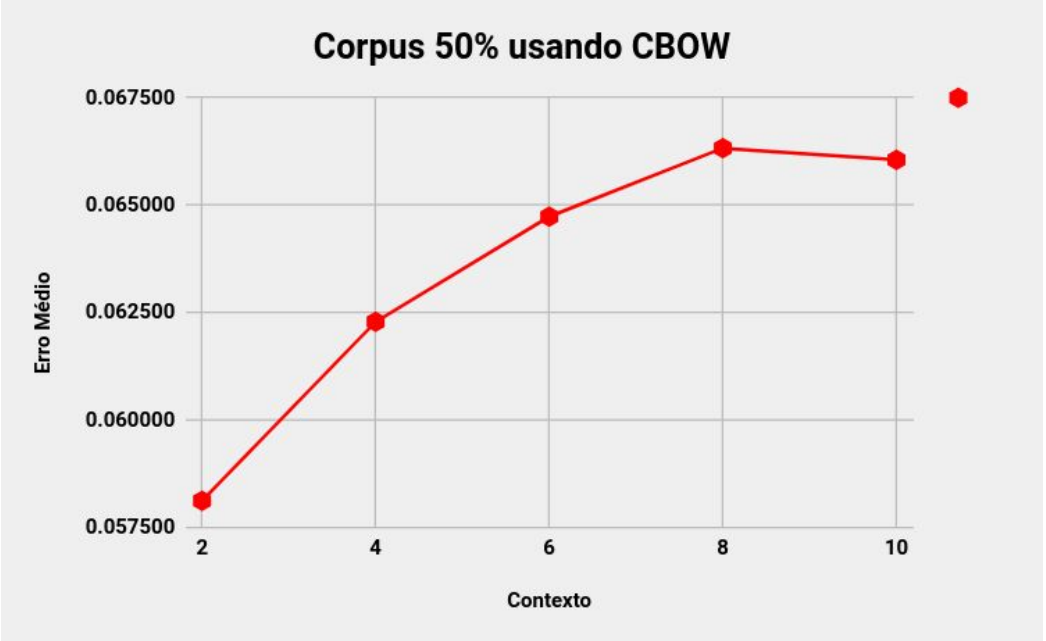
- **Modelo**: Descreve brevemente quais das três variáveis definidas (tamanho do corpus em porcentagem, tamanho do contexto, algoritmos skip-gram (1) ou CBOW (0)) foram usados em um dado modelo, respectivamente.
- **Corpus:** Descreve a porcentagem utilizada do corpus para um dado modelo.
- **Contexto:** Descreve o tamanho do contexto utilizado para um dado modelo.
- **Skip-gram**: Descreve qual algoritmo foi utilizado para treinar o modelo, sendo 1 para skip-gram e 0 para CBOW.
- **Erro**: Descreve a média das diferenças entre as distâncias da palavra no topo do ranking e a palavra correta, para todas as palavras testadas, por modelo.
- **Acertos**: Descreve a porcentagem total de acertos de predição da quarta palavra, dadas outras três como contexto, utilizando o arquivo *questions-words.txt.*

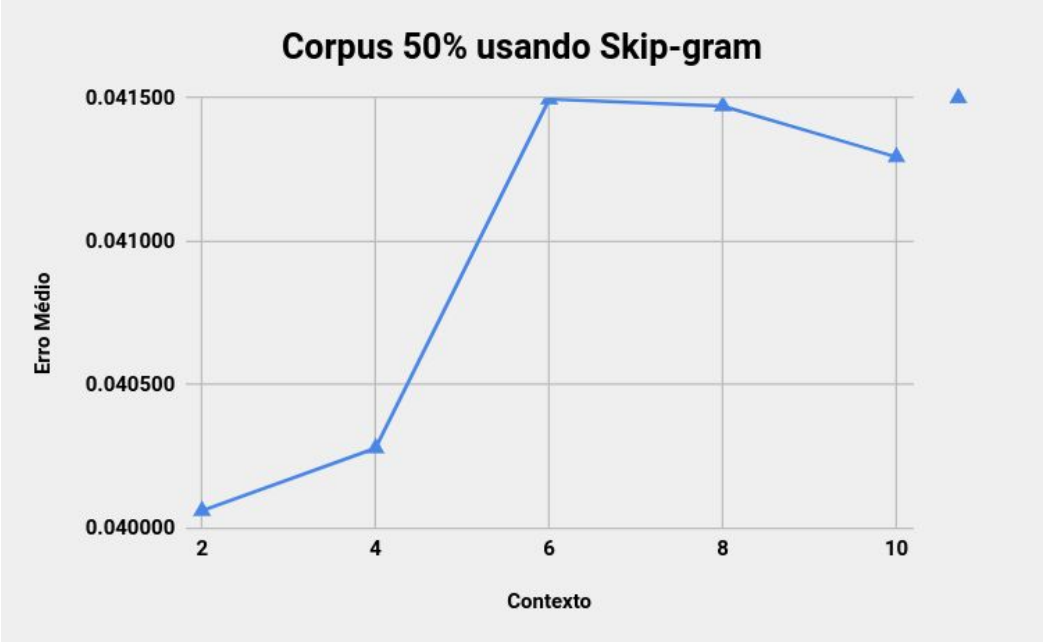
Como é possível observar na Tabela 1, o algoritmo skip-gram tem um resultado consistentemente melhor em relação ao CBOW, tanto para a porcentagem de acertos quanto para a média dos erros obtidos, considerando modelos com corpus e contextos de mesmo tamanho.

3.2 Gráficos de Linha

A seguir, gráficos representando as médias de erros para cada tamanho de corpus, divididos por algoritmo (skip-gram e CBOW):

3.2.1 Corpus 50%

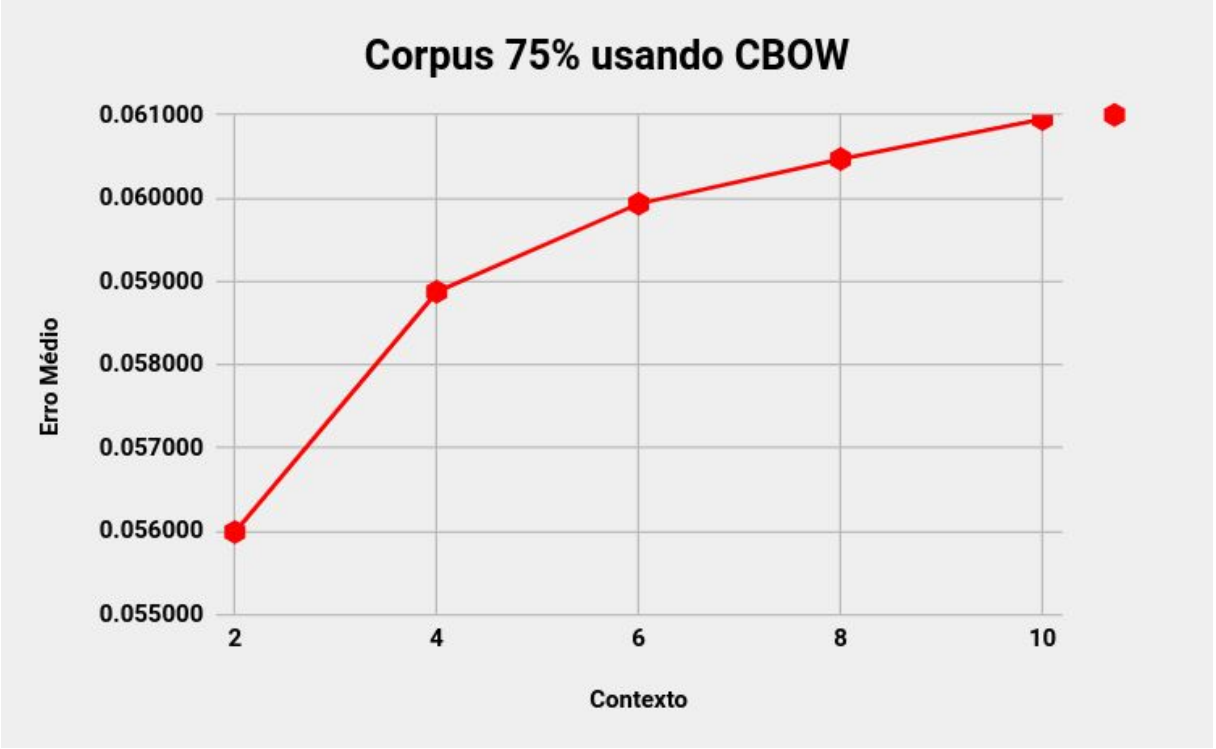


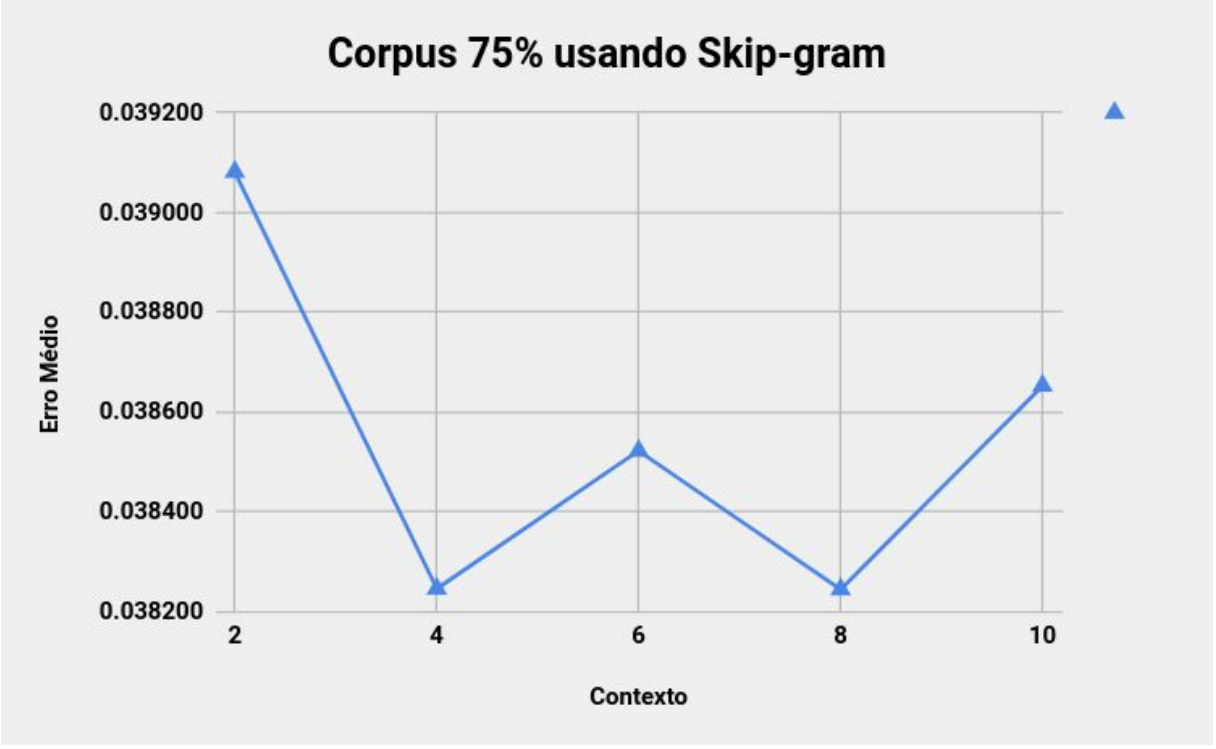


Como é possível observar nos gráficos acima, apesar do modelo utilizando skip-gram obter menores valores para erro médio, os dois modelos apresentam comportamentos parecidos, com crescimento entre contexto de tamanhos 2 e 4, e

um leve declínio entre os contextos de tamanho 8 e 10, considerando um corpus com 50% do tamanho original.

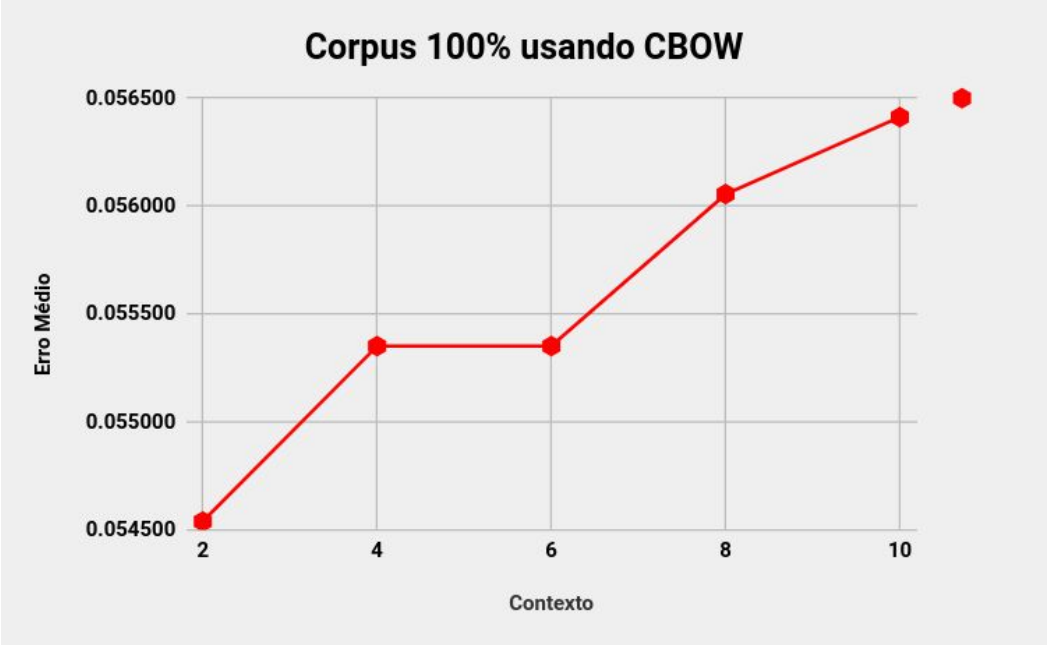
3.2.2 Corpus 75%

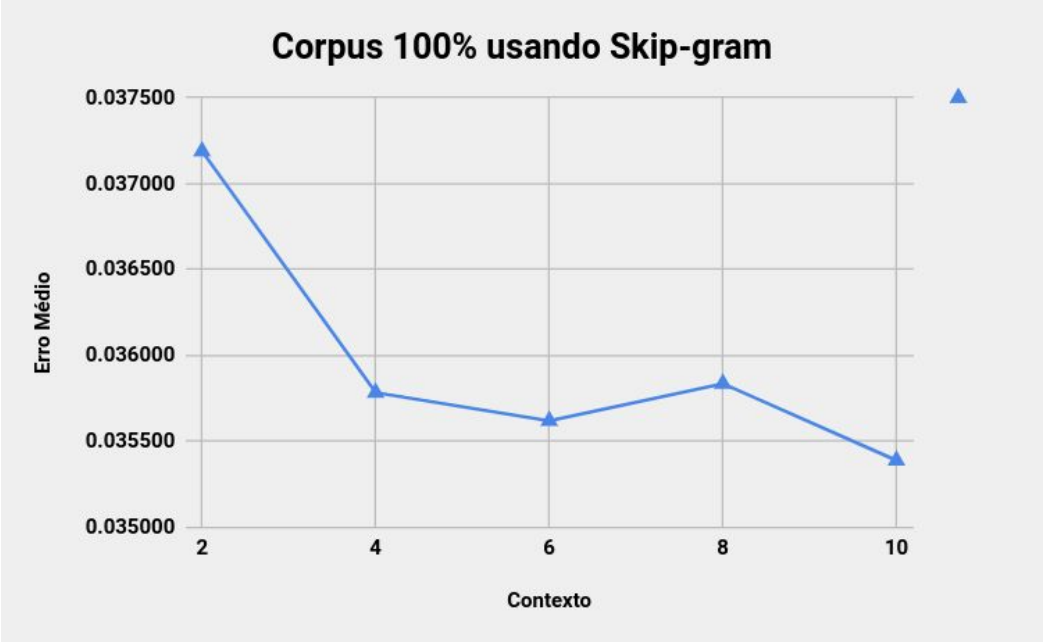




Utilizando corpus de tamanho 75% é possível notar um comportamento mais distinto, comparando os dois algoritmos utilizados. Novamente, os valores para erro médio utilizando skip-gram são significantemente menores que o encontrados utilizando CBOW. Utilizando skip-gram os valores tendem a diminuir, considerando o intervalo de contextos considerado. Já com CBOW, o erro médio tende a subir conforme o tamanho do contexto aumenta.

3.2.3 Corpus 100%





Considerando 100% do corpus, os valores para erro médio obtidos utilizando skip-gram continuam menores que os obtidos utilizando CBOW. Além disso, utilizando o corpus maior, é possível observar que, ao progressivamente utilizar

contextos maiores, os modelos que usam skip-gram obtêm valores menores para erros médios, ao passo que os modelos que utilizam CBOW obtêm erros médios maiores, conforme o tamanho do contexto cresce, algo que já é possível observar na seção 3.2.2, mas que tem comportamento mais distinto ao utilizar um corpus de tamanho maior.

4. Conclusão

Como foi possível observar na Seção 3, todos os modelos com menores valores para erro médio e maior porcentagem de acertos (relativa ou absoluta) utilizavam o algoritmo skip-gram. Esse resultado é condizente com a forma em que os dois algoritmos operam, visto que o skip-gram funciona melhor quando há menor diversidade de contextos disponível no corpus[6].

Referências

[1] http://mattmahoney.net/dc/text8.zip
[2] https://radimrehurek.com/gensim/
[3] https://code.google.com/archiv#e/p/word2vec/source/default/source
[4] https://radimrehurek.com/gensim/
[5] https://rare-technologies.com/word2vec-tutorial/
[6] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems. 2013.