Entropia e Geração de Séries de Aproximação utilizando uma ferramenta JAVA

Leopoldo G. N. Rabelo, Renato M. de Moraes

***Resumo* – Este artigo descreve uma ferramenta de código aberto desenvolvida na linguagem de programação JAVA que obtém o cálculo aproximado da entropia de uma língua natural através do uso de textos em formato digital da língua em questão. Nossa ferramenta também permite automatizar o processo de geração de séries de aproximação da linguagem, seja pela utilização de *n*-gramas, seja por fontes de Markov. Foram reproduzidos os resultados de Claude E. Shannon para a língua inglesa a título de validação, e em seguida, foram obtidos os resultados para a língua portuguesa. A ferramenta desenvolvida e aqui descrita pode ser utilizada tanto didaticamente no ensino da disciplina Teoria da Informação, como pode também ser empregada para estudos mais gerais de codificação de fonte, cálculo de entropia, compressão de dados, como também em estudos de estrutura da linguagem.**

***Palavras Chave*— Entropia, estrutura da linguagem, JAVA, séries de aproximação, Teoria da Informação.**

***Abstract* – This paper describes an open source tool developed in JAVA programming language which permits to calculate the approximate entropy value of a natural language using texts in digital format. Our tool also automatically generates series of approximations to a natural language either by using *n*-grams, or by employing Markov sources. The Claude E. Shannon's results for English were reproduced in order to validate our approach, as well as we obtained results for Portuguese. Our developed tool can help on lessons of Information Theory, and it can be utilized for more general studies of source coding, data compression, and in studies of structure of language.**

***Keywords*— Entropy, Information Theory, JAVA, series of approximation, structure of language.**

I. INTRODUCÃO

Aumenta a cada dia a preocupação com a transmissão de dados, tanto pela quantidade de informação gerada cada vez maior, quanto pela segurança da informação. Dessa forma são necessárias técnicas e ferramentas que permitam fazer um melhor uso do canal de transmissão. Técnicas de codificação de fonte ou de compressão de dados surgiram [1] para representar a informação com o mínimo de redundância possível, de modo a conseguir uma economia na banda de transmissão ou redução da quantidade de espaço necessário para armazenamento digital. Já técnicas de proteção da informação, como a criptografia, garantem a segurança das informações [2]. Em ambos os casos é preciso ter um conhecimento sobre como essa informação é gerada e como ela pode ser medida.

Em 1948 Claude E. Shannon propôs o modelo teórico matemático para a comunicação [3], que a concebe como uma transmissão de sinais. É um modelo linear visto como um processo de transporte da informação de um emissor para um receptor, no qual as mensagens são representadas por sinais a serem decodificados por um receptor. O transmissor pode ser modelado como uma fonte de informação que gera símbolos discretos. Estes símbolos podem estar associados a vários significados. Por exemplo, um dos estudos de Shannon foi considerar a fonte de informação emitindo mensagens compostas de palavras na língua inglesa, um gerador de textos, e propôs modelos que se aproximam do processo físico de geração da informação [3]. Porém, as técnicas e o modelo desenvolvidos podem ser aplicados a qualquer linguagem natural, inclusive aquelas, cujos alfabetos diferem do ocidental, como Árabe ou Mandarim e ainda para linguagens artificiais desde que elas possuam um conjunto básico de símbolos, ou alfabeto, e que cada símbolo tenha uma probabilidade associada a sua ocorrência, como o Esperanto.

Por outro lado, é importante o conhecimento da estrutura da linguagem quando se está envolvido em atividades como a compactação, a codificação e a proteção de dados. O estudo de Shannon estabeleceu uma série de técnicas sobre fontes geradoras de caracteres seguindo regras da língua inglesa.

Neste artigo demonstramos como algumas destas técnicas podem ser automatizadas através de um programa de código aberto realizado na linguagem de programação JAVA [4] que, além de outras funções, calcula as probabilidades das letras do alfabeto e a entropia de uma dada língua a partir de uma obra digitalizada, por exemplo, um livro em formato texto (.txt). Além disso, nosso programa é capaz de gerar séries de aproximação dessa língua também a partir do texto fornecido, seja pelo método de *n*-gramas, seja por fontes de Markov (ou gerador Markoviano) [5]. Estas técnicas de geração de informação são também conhecidas como geradores ou fontes de texto.

O programa executável está disponível na Internet [6] para uso público e pode ser utilizado didaticamente para auxiliar no ensino da disciplina de Teoria da Informação, como também, pode ajudar a outros pesquisadores interessados no assunto que porventura necessitem de uma ferramenta fácil e ágil, daí a nossa escolha pela linguagem de programação

L. G. N. Rabelo e R. M. de Moraes estão no Departamento de Sistemas e Computação (DSC), Escola Politécnica, Universidade de Pernambuco, Recife, Brasil. E-mails: leopoldorabelo@gmail.com, renato@dsc.upe.br.

Este trabalho foi financiado em parte pela Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE), pela Escola Politécnica de Pernambuco, e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

JAVA [4]. O código fonte de nossa ferramenta também está disponível em [6] de forma que a comunidade pode modificá-lo, adaptá-lo ou até mesmo melhorá-lo. Outro aspecto que motivou nosso estudo foi produzir resultados para a língua portuguesa análogos ao que Shannon produziu para o Inglês.

O restante deste trabalho está organizado da seguinte forma. A Seção II faz uma breve revisão sobre os conceitos básicos de Teoria da Informação relacionados ao nosso estudo. A Seção III apresenta a lógica de funcionamento da ferramenta JAVA e explica como as funções desenvolvidas no trabalho de Shannon para séries de aproximação foram realizadas. A Seção IV descreve como foram realizados os testes e quais foram os resultados obtidos para as línguas inglesa e portuguesa. A Seção V conclui o trabalho, resumindo suas principais contribuições e trabalhos futuros.

II. Conceitos básicos

A gramática, ou o conjunto de regras de uma linguagem natural como o Inglês ou Português, definem como os símbolos desta linguagem podem ser empregados para formar um texto. Os caracteres ou letras desta linguagem são os símbolos elementares e ao conjunto total destes caracteres denomina-se alfabeto. Por exemplo, o alfabeto da língua inglesa é formado pelo seguinte conjunto de símbolos ou caracteres {a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z}. Uma série de n letras sobre este alfabeto é dita ser um n-grama ou série de grau n sobre o alfabeto [3], [5]. O dicionário da língua inglesa especifica como estes símbolos podem ser reunidos para formarem palavras que tenham um sentido, ou significado. A estes símbolos, adiciona-se o espaço (ou intervalo, doravante representado por "_") que serve para separar as palavras. As palavras separadas pelos espaços podem ser reunidas para formarem textos que seguem as regras gramaticais da língua em questão. Estas regras são complexas para as linguagens naturais e não seria matematicamente viável um modelo que reproduzisse seu comportamento com exatidão. Entretanto, Shannon propôs um procedimento para geração de séries, ou textos, que se aproxima da linguagem natural conforme iremos descrever ao longo deste trabalho.

Um conceito importante neste contexto é a entropia de uma fonte geradora de símbolos. Seja uma fonte sem memória que emite símbolos contido em um alfabeto finito $S=\{s_1, s_2, ..., s_n\}$, onde os símbolos ocorrem com probabilidades $P(s_1)$, $P(s_2)$, ..., $P(s_n)$, tal que $\sum_{i=1}^{n} P(s_i)=1$. Então, a entropia desta fonte é definida por [3], [5]:

$$H_r(S) = \sum_{i=1}^{n} P(s_i) \log_r \frac{1}{P(s_i)} \text{ unidades } r\text{-árias/símbolo, (1)}$$

onde r é a base do logaritmo e determina a unidade de medida da entropia. Por exemplo, na base binária, $r=2$, a unidade da entropia é dada em bits por símbolo [3]. A entropia de uma fonte de informação determina, na unidade r-ária, o comprimento mínimo que o número médio de símbolos de um código, contendo r símbolos no alfabeto-código, pode ter para representar um símbolo desta fonte [3], [5].

III. Descrição da Ferramenta Java

Nesta seção é apresentada a ferramenta JAVA [4], [6], tendo suas funcionalidades e características explicadas. Será descrito o processo que permite, a partir de um texto escrito em uma linguagem natural, gerar séries de aproximação e calcular sua entropia. A geração das séries e o cálculo da entropia requerem o conhecimento das probabilidades e estas são obtidas a partir da freqüência de ocorrência dos símbolos no texto. De forma geral o programa permite: (i) processar um texto oriundo de uma linguagem natural, no caso Inglês ou Português, contendo milhares de palavras, provavelmente um livro; (ii) calcular a partir deste texto, a freqüência relativa de ocorrência de cada letra, definida no alfabeto de símbolos da linguagem e assim determinar a probabilidade de cada letra e de séries de letras; (iii) calcular, a partir das probabilidades encontradas, a entropia do texto em questão; (iv) gerar séries de aproximação para a língua associada ao texto empregado, seja pela utilização de n-gramas, seja pelo uso de fontes de Markov [3].

O texto é o ponto de partida do processo de geração de séries e do cálculo da entropia. Note que um texto em Português contém uma variedade de símbolos maior que um texto em inglês devido a existência no Português de palavras acentuadas e do "ç". A fim de realizar uma comparação com o estudo feito para a língua inglesa por Shannon [3], foi considerado apenas o alfabeto de vinte e seis letras e o espaço, dessa forma realizamos algumas alterações nos textos antes de processá-los. As letras que num texto em português se apresentam acentuadas foram convertidas e contabilizadas sem o acento, ou seja, um "á" por exemplo, é considerado simplesmente como um "a", o "ç" é transformado em "c" e o hífen é considerado como um espaço. Foram ainda removidos os caracteres de pontuação, como também os números, similar ao que foi realizado por Shannon [3]. O programa ainda converte letras minúsculas em maiúsculas com o objetivo de manter um padrão na geração das séries de aproximação [3], [5]. Portanto, nossa ferramenta JAVA desconsidera quaisquer outros símbolos que não estão contidos no alfabeto mais o espaço. Porém, para uma análise mais exata da estrutura de uma dada língua, basta considerar todos os símbolos que nela acontecem e modificar apropriadamente a ferramenta disponível em código aberto em [6].

A. Obtenção das Probabilidades e Entropia

A fase de obtenção das probabilidades ocorre após o texto ser lido pelo programa e este ser processado de forma que os caracteres indesejados, isto é, aqueles que não fazem parte do alfabeto mais o espaço, são convertidos ou eliminados. O usuário então tem duas opções de cálculo das probabilidades: o cálculo das probabilidades das letras e o cálculo das probabilidades das palavras. A opção de cálculo por letras da ferramenta inicia ao mesmo tempo o cálculo das probabilidades das letras e dos n-gramas até o grau cinco. Desta forma, as probabilidades serão disponibilizadas para o usuário após o termino de processamento de todos os n-gramas.

A Figura 1 ilustra a tela principal da ferramenta JAVA desenvolvida com as probabilidades calculadas a partir do

arquivo texto do famoso poema "*The Raven*" do autor americano Edgar Allan Poe. As probabilidades estão mostradas em valores percentuais.

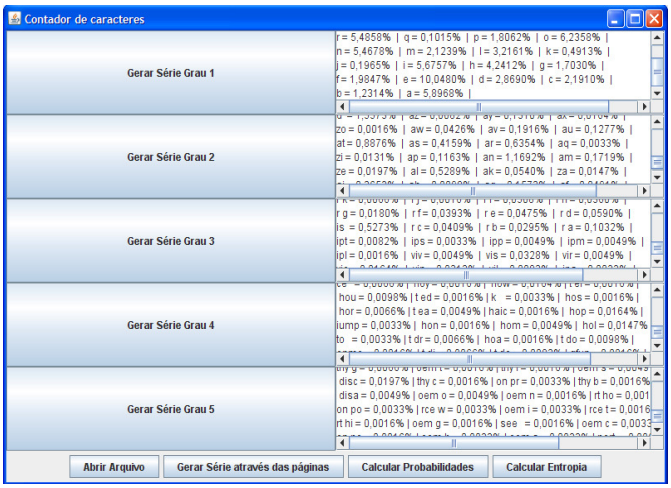

Fig. 1. Tela da ferramenta JAVA com as probabilidades calculadas a partir do poema intitulado "*The Raven*" do autor Edgar Allan Poe.

O cálculo das probabilidades das letras é dado pela razão do número de ocorrências de uma determinada letra pelo número total de símbolos do alfabeto contido no texto, isto é, sua freqüência relativa. Portanto, quanto mais longo for o texto em análise, ou seja, quanto mais caracteres ele contenha, mais a freqüência relativa de cada símbolo se aproxima de sua probabilidade [3], [5].

Para os *n*-gramas existem duas interpretações. A primeira é formar *n*-gramas através da combinação dos caracteres de uma palavra, ou seja, é possível formar $(x-n+1)$ *n*-gramas a partir de uma palavra, onde x é o número de caracteres da palavra e n é o grau do *n*-grama e $x \geq n$. Por exemplo, na série de caracteres "cabana" é possível formar seis monogramas (c,a,b,a,n,a), cinco digramas (ca,ab,ba,an,na), quatro trigramas (cab,aba,ban,ana), três tetragramas (caba,aban,bana), dois pentagramas (caban,abana) e um hexagrama (cabana). Obviamente não é possível formar um heptagrama, já que esse grau é maior que numero de caracteres presentes na palavra.

A outra interpretação é dividir as palavras de modo a formar os *n*-gramas, ou seja, uma palavra com seis caracteres formaria seis monogramas, três digramas e dois trigramas.

Adotamos a primeira interpretação por considerá-la mais abrangente já que permite a formação de mais *n*-gramas. Para essa interpretação o cálculo das probabilidades ocorre contabilizando a ocorrência dos *n*-gramas e dividindo pelo total de *n*-gramas gerados. O nosso programa efetua o cálculo até o pentagrama.

Nos resultados da próxima seção, os espaços são exibidos junto com as palavras, uma vez que estes estão incluídos no conjunto possíveis de caracteres, e servirão para separar as palavras no momento da geração das séries.

A entropia pode ser obtida utilizando as probabilidades dos *n*-gramas ou mesmo das palavras, conforme a Equação (1) considerando-se os símbolos de acordo com o caso em questão, isto é, o símbolo pode ser o monograma para ordem um, o digrama para ordem dois, etc. Observe que neste caso, o que se está calculando na verdade são estimativas dos valores de entropia já que os modelos de ordem *n* são aproximações da linguagem natural [3], [5]. A opção de calcular a entropia somente será disponibilizada pelo programa após o cálculo das probabilidades.

Outra opção indicada por Shannon [3] e Abramson [5] é o cálculo das probabilidades das palavras contidas no texto, ao invés dos *n*-gramas. Nosso programa realiza esse cálculo de forma análoga ao que foi realizado para as letras, considerando como palavra uma série de caracteres delimitada pela presença de espaços. A opção por palavras calcula ao mesmo tempo a ocorrência das palavras isoladamente e a ocorrência de duas palavras e três palavras seguidas, de forma similar aos *n*-gramas. Assim, a série formada de uma palavra mais espaço e mais outra palavra é armazenada com um valor associado, o qual indica a ocorrência dessa série e é incrementado a cada ocorrência de uma série idêntica. O método utilizado no programa forma todas as séries possíveis de acordo com o grau em questão. Por exemplo, para o segundo grau, uma série como "casa de madeira" gera duas séries possíveis, uma formada por "casa de" e outra formada por "de madeira". Esta idéia se estende para os graus maiores.

B. Séries de Aproximação

As séries de aproximação por letras podem ser obtidas de duas maneiras diferentes: a geração por *n*-gramas e a geração por fontes de Markov, e têm como objetivo gerar textos que se aproximam da linguagem natural em questão. Há um terceiro método de geração de textos partindo-se de palavras já formadas usando-se suas freqüências relativas [3], [5].

A opção de gerar as séries de aproximação por letras é disponibilizada pelo programa após o cálculo das probabilidades de todos os *n*-gramas até o grau cinco. O primeiro caso, ou 1-grama, gera séries emitindo um símbolo de cada vez considerando a probabilidade de cada letra calculada pelo programa a partir do texto utilizado. A saída de cada símbolo ocorre independente da saída anterior. O processo de geração de séries segue a seguinte lógica: (i) o usuário determina qual série quer gerar dentre as cinco opções de 1-grama a 5-grama.[1] As cinco opções de séries são disponibilizadas ao mesmo tempo; (ii) o usuário escolhe a quantidade total de caracteres que serão gerados; (iii) o programa cria uma matriz cujo tamanho é o somatório das freqüências obtidas pelo cálculo de probabilidade e preenche essa matriz com a quantidade relativa a cada probabilidade dos *n*-gramas; (iv) uma função geradora de números aleatórios uniforme, determina o símbolo a ser gerado pela fonte de texto. A função gera números contidos em um intervalo igual ao tamanho da matriz. O número na verdade indica a posição da matriz. O valor contido nesta posição será a letra gerada pela fonte; (v) a letra escolhida aleatoriamente é

[1] Observe que também existem series de grau zero onde a probabilidade de ocorrência de cada símbolo é independente e uniformemente distribuída [3], [5]. Ou seja, um gerador de ordem zero que emprega um alfabeto com 27 símbolos criará caracteres independentemente, cada um com igual probabilidade 1/27 de ocorrer. Optamos por não implementar esta opção na nossa ferramenta devido a trivialidade desse tipo de gerador.

exibida; (vi) o processo se repete até o tamanho de caracteres escolhido pelo usuário em (ii).

Por outro lado, a geração por fonte de Markov funciona da seguinte forma [3], [5]: (i) abre-se um livro numa página qualquer e escolhe-se aleatoriamente uma letra nesta página que em seguida fica armazenada na memória do programa; (ii) abre-se novamente o livro noutra página qualquer e lê-se a página até se encontrar a primeira ocorrência da letra igual à gravada na memória armazenando-se a letra que ocorre logo após na palavra; (iii) o processo a partir de (ii) repete-se para esta nova letra, e assim sucessivamente a série armazenada é uma seqüência de grau um desejada, pois se trata de uma fonte de Markov de primeira ordem. Para séries de graus maiores, basta se considerar a *n*-ésima letra seguinte após sua próxima ocorrência.

A lógica do gerador Markoviano foi empregada no nosso programa da seguinte maneira: (i) o usuário escolhe a opção de gerar séries através das páginas do livro; (ii) é dado ao usuário a opção de gerar séries de grau um até cinco, mas ao contrário da opção de gerar séries pelos *n*-gramas, apenas uma opção é disponibilizada de cada vez; (iii) se o usuário escolher outra opção que não as permitidas, uma mensagem de erro é exibida e a tela com as opções é novamente mostrada ao usuário; (iv) ao usuário, após este escolher corretamente dentre as opções válidas, é apresentado uma nova tela para a inserção do tamanho da seqüência de caracteres a ser gerada; (v) o programa então realiza o processo conforme descrito no parágrafo anterior. Uma função geradora de números aleatórios determina a página a ser aberta. Como o texto na prática é uma seqüência de caracteres, foi preciso definir o que seria uma página. A página foi definida como uma seqüência de cinqüenta linhas e com um total de dois mil caracteres; (vi) uma vez escolhida a página, tem-se novamente o processo de escolha aleatória, sendo agora para a letra. A letra é escolhida dentre as duas mil por uma função que gera números aleatórios uniformemente contidos em um intervalo de um a dois mil; (vii) o caractere escolhido é exibido e seu valor armazenado para posterior comparação; (viii) a mesma função do item (v) escolhe a nova página e a percorre a partir da posição inicial comparando os valores com o valor armazenado; (ix) se o valor for igual ocorre um incremento na posição e o valor dessa posição é exibido e ocupa o lugar do caractere que estava sendo usado para comparação; (x) o processo continua até que seja gerada uma série de tamanho igual ao determinado pelo usuário; (xi) se o caractere não for encontrado na página a busca continua na página seguinte até o final do livro; (xii) se a busca não obtiver resultado até o final, o processo retorna ao inicio do livro e recomeça da posição inicial do texto.

Este método está disponível ao usuário do programa imediatamente após o texto ser carregado, uma vez que não faz uso das probabilidades calculadas.

A última possibilidade para séries de aproximação da linguagem é a geração de seqüências baseadas nas probabilidades de ocorrência das palavras. O processo é análogo ao descrito para geração por letras e, portanto, o omitiremos para fins de economia de espaço neste artigo.

IV. Experimentos e Resultados

Os experimentos foram realizados em três partes. Na primeira, foi calculada a entropia e geradas seqüências de aproximação para a língua inglesa e os resultados foram comparados com aqueles obtidos por Shannon [3] e Abramson [5] de forma a verificar e validar as funcionalidades do programa desenvolvido. Após a validação da ferramenta, foram realizados experimentos para a língua portuguesa e os resultados foram comparados com aqueles obtidos por Silva Filho [7]. Na última parte comparamos o desempenho da ferramenta JAVA apresentando tempos de execução do programa para alguns exemplos.

A. Resultados para a Língua Inglesa

O objetivo dos experimentos com a língua inglesa foi validar nossa ferramenta, e para tal comparamos nossos resultados, obtidos pela execução do programa utilizando textos em língua inglesa, com os apresentados por Shannon [2] e Abransom [5]. As principais funções avaliadas foram o cálculo das probabilidades, a geração de séries de aproximação e o cálculo da entropia.

Como dito anteriormente, utilizamos diversos livros nos experimentos. Os livros foram obtidos em formato de texto digital (.txt) e estão disponíveis na biblioteca digital *Project Gutenberg* [8]. A seguir listamos os títulos e respectivos autores de cada texto utilizado nos experimentos para língua inglesa:

- *The Raven* – Edgar Allan Poe;
- *The Adventures of Huckleberry Finn* – Mark Twain;
- *The Wonderful Wizard of Oz* – Lyman Frank Baum.

A Tabela I apresenta o cálculo das probabilidades, em valores percentuais, para cada símbolo do alfabeto e o espaço. Para a nossa ferramenta JAVA foi utilizado o poema "*The Raven*" e para comparação estão apresentadas as probabilidades fornecidas por Abramson em [5].

TABELA I
Comparação das probabilidades para o Inglês obtidas pela ferramenta JAVA a partir do poema "*The Raven*" com as fornecidas por Abramson [5].

Símbolo	Probabilidade		Símbolo	Probabilidade	
	Ferramenta JAVA	Abramson		Ferramenta JAVA	Abramson
A	5,90%	6,42%	O	6,24%	6,32%
B	1,23%	1,27%	P	1,80%	1,52%
C	2,19%	2,18%	Q	0,10%	0,08%
D	2,87%	3,17%	R	5,49%	4,84%
E	10,05%	10,31%	S	4,86%	5,14%
F	1,98%	2,08%	T	7,59%	7,96%
G	1,70%	1,52%	U	2,30%	2,28%
H	4,24%	4,67%	V	0,78%	0,83%
I	5,68%	5,75%	W	1,46%	1,75%
J	0,19%	0,08%	X	0,11%	0,13%
K	0,49%	0,49%	Y	1,43%	1,64%
L	3,22%	3,21%	Z	0,05%	0,05%
M	2,12%	1,98%	espaço	22,60%	18,59%
N	5,47%	5,74%			

Observa-se que os valores obtidos são próximos e o único símbolo com diferença acima de 1% é o espaço devido ao fato do nosso programa contabilizar um espaço a cada vez que

encontra o final de uma linha no texto. Outra peculiaridade também observada é o fato de que a letra "E" seja o caractere mais freqüente da língua inglesa [5], fora o espaço, como já se deveria esperar.

A Figura 2 apresenta uma série de aproximação de grau um gerada pela nossa ferramenta a partir das probabilidades calculadas do poema "*The Raven*".

LNOT_PO_R_OWV_GOOD_TO_N__BATLTHEWGFE_
MH_RIHOIREAVTIET_O_OTF_MT_L_S_RRW_HUG_
MNOI_OL_YI_M_URTIMTP_U_

Fig. 2. Série de grau um gerada na ferramenta JAVA a partir das probabilidades dos caracteres obtidas do poema "*The Raven*".

Para fins de comparação, as Figuras 3 e 4 apresentam séries de aproximação de grau um obtidas por Abransom [5] e por Shannon [3], respectivamente.

URTESHETHING_AD_E_AT_FOULE_ITHALIORT_WACT_
D_STE_MINTSAN_OLINS_TWID_OULY_TE_THIGHE_
CO_YS_TH_HR_UPA

Fig. 3. Série de grau um gerada por Abramson [5].

O número de caracteres gerados é um parâmetro fornecido pelo usuário ao programa e na Figura 2 são mostrados 100 caracteres, que é compatível com o número de caracteres gerados por Abramson e Shannon. Também na Figura 2 é possível distinguir duas palavras da língua inglesa: GOOD, TO; enquanto que na Figura 3 também identificamos duas palavras: AD, AT; e na Figura 4 nenhuma palavra pode ser reconhecida. Portanto, observa-se que um gerador de grau um cria textos bastante distintos dos de uma linguagem natural, onde pouquíssimas palavras podem ser distinguidas a cada centena de caracteres gerados. Entretanto, já é possível perceber a estrutura da linguagem, pois as "palavras" geradas possuem comprimentos compatíveis com a língua inglesa e as proporções de vogais e consoantes se aproximam da realidade [5].

OCRO_HLI_RGWR_NMIELWIS_
EU_LL_NBNESEB_YA_TH_
EEI_ALHENHTTPA_OOBTTVA_NAH_BRL

Fig. 4. Série de grau um gerada por Shannon [3].

Por outro lado, à medida que o grau do gerador aumenta, o número de palavras distinguíveis cresce conforme pode ser observado nas Figuras 5 e 6 que ilustram séries de aproximação de grau três, contendo 73 caracteres cada, obtidos agora pelo gerador Markoviano na nossa ferramenta, a partir da obra "*The Adventures of Huckleberry Finn*", e em Abramson [5], respectivamente.

ANK_ING_ALLAZED_OF_THE_DON_AND_WAY_SH_
ITHE_HIS_BECK_OVER_NE_NOOD_HE_A_F

Fig. 5. Série de grau três gerada na ferramenta JAVA pelo gerador Markoviano a partir da obra "*The Adventures of Huckleberry Finn*".

Na Figura 5 pode-se identificar nove palavras: OF, THE, AND, WAY, HIS, BECK, OVER, HE, A, enquanto na Figura 6 é possível também distinguir nove palavras: CAN, OF, TO, OF, TO, A, MAND, AND, BUT. Note que, para estes casos, o número de palavras reconhecíveis aumentou como era de se esperar mesmo reduzindo o número de caracteres gerados, já que se trata de um melhor modelo de aproximação.

IANKS_CAN_OU_ANG_RLER_THATTED_OF_TO_S HOR
_OF_TO_HAVEMEM_A_I_MAND_AND_BUT_

Fig. 6. Série de grau três obtida pelo gerador Markoviano em Abramson [5].

A fim de continuar a validação da nossa ferramenta, comparamos várias outras situações com nosso programa e com os resultados de Shannon [3] e de Abramson [5], por exemplo, para séries de graus quatro e cinco, onde constatamos resultados análogos, porém preferimos omiti-los dada a falta de espaço disponível neste artigo e pelo fato de que queremos mostrar alguns exemplos para a língua portuguesa.

Para finalizar, apresentamos na Tabela II a comparação entre os valores referente às estimativas dos limites superiores da entropia da língua inglesa realizados pela nossa ferramenta, a partir da obra "*The Wonderful Wizard of Oz*", e os disponibilizados por Shannon em [9], para os diferentes graus de aproximação, onde constatamos valores similares até a ordem cinco.

TABELA II
Comparação dos valores de estimativas de entropia para a língua inglesa obtidos pela ferramenta JAVA e os fornecidos por Shannon [9].

Ferramenta JAVA		Shannon	
Ordem	Entropia (bits/símbolo)	Ordem	Entropia (bits/símbolo)
1ª	4,04	1ª	4,03
2ª	3,66	2ª	3,32
3ª	3,27	3ª	3,10
4ª	2,92	4ª	2,6
5ª	2,63	5ª	2,7

B. Resultados para a Língua Portuguesa

O objetivo desta subseção é apresentar alguns resultados para a língua portuguesa seguindo o procedimento descrito anteriormente. Os livros foram obtidos em formato de texto digital (.txt) e estão disponíveis na biblioteca digital Domínio Público [10]. Nossos resultados foram comparados com Silva Filho [7] que também realizou estudos de séries de aproximação para a língua portuguesa. A seguir listamos os títulos e respectivos autores de cada texto utilizado nos experimentos para língua portuguesa:

- A Escrava Isaura – Bernardo Guimarães;
- Dom Casmurro – Machado de Assis;
- Os Lusíadas – Luis Vaz de Camões.

A Tabela III apresenta os valores referentes ao cálculo das probabilidades, em valores percentuais, para língua portuguesa e sua comparação com os valores apresentados por Silva Filho [7]. Os valores obtidos pela nossa ferramenta são provenientes da obra "Dom Casmurro". Mais uma vez observamos boa concordância de valores, exceto pela diferença do espaço conforme já explicado. Uma peculiaridade a se notar na língua portuguesa é o fato de a letra "A" ser a mais freqüente, enquanto que as letras "K", "Y" e "W" possuem probabilidades de ocorrência

praticamente nulas. Interessante também é a baixa freqüência da letra "J" constatada por ambos os casos.

TABELA III
Probabilidades para o Português obtidas pela ferramenta JAVA a partir da obra "Dom Casmurro" e as fornecidas por Silva Filho [7].

Símbolo	Probabilidade		Símbolo	Probabilidade	
	Ferramenta JAVA	Silva Filho		Ferramenta JAVA	Silva Filho
A	12,14%	11,40%	O	8,52%	8,00%
B	0,70%	0,80%	P	2,22%	2,40%
C	3,01%	4,50%	Q	1,20%	0,70%
D	3,79%	4,60%	R	4,96%	5,30%
E	10,74%	11,40%	S	6,43%	7,10%
F	0,80%	1,00%	T	3,43%	4,60%
G	0,89%	1,10%	U	4,02%	3,00%
H	1,09%	0,60%	V	1,41%	1,20%
I	5,26%	5,60%	W	0,00%	0,00%
J	0,27%	0,20%	X	0,23%	0,20%
K	0,00%	0,10%	Y	0,00%	0,00%
L	2,20%	2,00%	Z	0,37%	0,30%
M	4,27%	4,20%	espaço	19,47%	15,60%
N	3,91%	4,10%			

A Figura 7 apresenta uma série de aproximação com 102 caracteres obtida por um gerador de Markov de ordem dois a partir da obra "A Escrava Isaura", onde podemos identificar as seguintes palavras: ESTOU, DE, DA, SOM, NÃO, EU, AS.

ESTOU_DE_ESPERENTRENISCIDAO_DA_SOM_NAO_
QUAR_UMARIMPLICABEM_EU_VIS_UMAISSAS_
LEIDAR_AS_ININHO_CURAZENC

Fig. 7. Série de grau dois obtida na ferramenta JAVA pelo gerador Markoviano a partir da obra "Escrava Isaura".

A Figura 8 mostra uma série de aproximação com 103 caracteres obtida por Silva Filho [7] também usando um gerador Markoviano de ordem dois, onde é possível identificar as palavras: AS, A, OU, AS.

AS_CRICA_METRIPTO_A_DEAL_SISOMOS_
TITIOMENTECIO_REAMEDIM_QUATUICANDO_
US_EXIS_OU_ATORMACAO_AS_PORDDAGEM

Fig. 8. Série de grau dois obtida por gerador Markoviano em Silva Filho [7].

Para calcular a entropia na língua portuguesa nossa ferramenta usou a obra "Dom Casmurro" e os valores obtidos são comparados com Silva Filho na Tabela IV, onde percebemos concordância nos resultados até a segunda ordem conforme disponibilizado em [7].

TABELA IV
Comparação dos valores de estimativas de entropia para a língua portuguesa obtidos pela nossa ferramenta e os fornecidos por Silva Filho [7].

Ferramenta JAVA		Silva Filho	
Ordem	Entropia (bits/símbolo)	Ordem	Entropia (bits/símbolo)
1ª	3,94	1ª	3,97
2ª	3,56	2ª	3,53
3ª	3,27	3ª	-
4ª	3,00	4ª	-
5ª	2,76	5ª	-

C. Desempenho da Ferramenta JAVA

Para fins de avaliação de desempenho apresentamos, na Tabela V, os tempos de execução do programa para calcular as probabilidades dos caracteres para alguns textos utilizados. Os tempos mostrados foram obtidos num computador usando processador Intel Core 2 Duo de 2,4 GHz, memória RAM de 4 Gbytes, com sistema operacional Windows XP. As variações nos tempos de execução do programa guardam proporcionalidade com o tamanho do arquivo da obra em análise. Ou seja, quanto maior o tamanho do arquivo, maior o número de caracteres a serem processados, resultando, portanto num tempo de execução maior.

TABELA V
Tempo de processamento para cálculo das probabilidades dos *n*-gramas.

Obra literária	Tempo de execução (minutos)
A Escrava Isaura	15
Os Lusíadas	16
Dom Casmurro	18
The Adventures of Huckleberry Finn	26
The Raven	4
The Wonderful Wizard of Oz	9

V. CONCLUSÃO

Este trabalho descreve uma ferramenta JAVA que permite calcular a entropia e gerar séries de aproximação de uma dada linguagem natural, no caso aquelas que usam o alfabeto contendo 26 letras mais o espaço. Foram reproduzidos alguns dos resultados obtidos por Claude E. Shannon para a língua inglesa, como também para o Português.

A ferramenta está disponível em código aberto na Internet para ser utilizada pela comunidade interessada nos assuntos relacionados à Teoria da Informação e estudos de estrutura da linguagem, e pode, portanto ser modificada para outros casos de interesse.

Como trabalhos futuros, pretendemos aperfeiçoar a ferramenta para que esta aceite dicionários de outras línguas, como Árabe, Mandarim ou mesmo Esperanto, e otimizar seu código JAVA no intuito de diminuir o tempo médio de execução do programa.

REFERÊNCIAS

[1] SAYOOD, K. Introduction to Data Compression. Morgan Kaufmann, 2000.
[2] TRAPPE, W.; WASHINGTON, L. C. Introduction to Cryptography with Coding Theory. Prentice Hall, 2002.
[3] SHANNON, C. E. The Mathematical Theory of Communication. *Bell System Technical Journal*, vol. 27, p. 379-423, July 1948.
[4] HORSTMANN, C. S.; CORNELL, G. Core Java (TM) 2, Vol. I – Fundamentals. Prentice Hall, 2004.
[5] ABRAMSON, N. Information Theory and Coding. McGraw-Hill Book Company, 1963.
[6] http://www.dsc.upe.br/~rmm/gerador_de_texto/index.html
[7] SILVA FILHO, J. G. Informação, Codificação e Segurança de Dados, UNB-FT-ENE, Brasília, 1998.
[8] Gutenberg Project, http://gutenberg.org/wiki/Main_Page
[9] SHANNON, C. E. Prediction and Entropy of Printed English, *Bell System Technical Journal*, vol. 30, p. 50-64, January 1951.
[10] Domínio Público, http://www.dominiopublico.gov.br/