INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE – IFRN

CLARA YASMIN CUNHA FERNANDES DOS SANTOS
PABLO DEYVID DE PAIVA

WHOAUTHOR: ATRIBUIDOR DE AUTORIA DE TEXTOS LITERÁRIOS UTILIZANDO INTELIGÊNCIA COMPUTACIONAL COM BASE NO PPM-C

PAU DOS FERROS

2021

CLARA YASMIN CUNHA FERNANDES DOS SANTOS
PABLO DEYVID DE PAIVA

WHOAUTHOR: ATRIBUIDOR DE AUTORIA DE TEXTOS LITERÁRIOS UTILIZANDO INTELIGÊNCIA COMPUTACIONAL COM BASE NO PPM-C

Trabalho de Conclusão de Curso apresentado ao Curso de Técnico Integrado em Informática do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial à obtenção do título de Técnico em Informática.
Orientador: Prof. Me. Irlan Arley Targino Moreira
Coorientador: Prof. Me. Elenilson Vieira da Silva Filho e Prof. Dr. Francisco Magno Silva de Araújo

PAU DOS FERROS

2021

Clara Yasmin Cunha Fernandes dos Santos
Pablo Deyvid de Paiva

WhoAuthor: Atribuidor de autoria de textos literários utilizando inteligência computacional com base no PPM-C

Trabalho de Conclusão de Curso apresentado ao Curso de Técnico Integrado em Informática do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial à obtenção do título de Técnico em Informática.

Prof. Me. Irlan Arley Targino Moreira
Orientador

Prof. Me. Elenilson Vieira da Silva Filho
Coorientador

Prof. Dr. Francisco Magno Silva de Araújo
Coorientador

Prof. Dr. Aluisio Igor Rego Fontes
Examinador interno

Pau dos Ferros
2021

AGRADECIMENTOS

CLARA YASMIN CUNHA FERNANDES DOS SANTOS

É com o coração repleto de gratidão que dedico esse trabalho a Deus, por até aqui ter me sustentado em amor e por ter abundado em minha vida suas misericórdias. Certamente, sem o Senhor, não estaria aqui.

Em seguida, agradeço ao meu pai, Claudener Fernandes, por ter me direcionado em minhas escolhas e garantido que, mesmo que eu caísse, não me machucasse. Também agradeço à minha mãe, Célia Fernandes, que tantas vezes me ouviu e aconselhou, pacientemente, abraçando-me e fortalecendo-me com as mais diversas demonstrações de amor e cuidado. Sou grata, ainda, à minha irmã, Cínthia Coeli, por ter me ensinado a ser forte e dedicar minha atenção às coisas que realmente importam.

Agradeço aos orientadores Irlan Arley, Elenilson Vieira e Francisco Magno, por aceitarem conduzir o trabalho, por todas as correções, palavras de incentivo e pela compreensão; os senhores foram indispensáveis para essa realização.

Ao meu parceiro de trabalho Pablo Deyvid, por me inspirar com seu talento e proatividade, impulsionando-me a dar sempre o meu melhor. Sou grata por todas as experiências dos últimos dois anos com alguém tão capaz como você ao meu lado.

Aos meus amigos, Verônica Rodrigues, Natan Felipe e Maria Cecília, obrigada por acreditarem no meu potencial e por serem como oásis no deserto para mim.

Por fim, sou grata ao IFRN, por ter proporcionado o ambiente e as condições para o desenvolvimento do projeto.

PABLO DEYVID DE PAIVA

Gostaria de agradecer e dedicar esse trabalho de TCC as seguintes pessoas:

À minha mãe, Zenia Holanda, ao meu pai, Leilton Paiva, e ao meu irmão, Lucas Tiago, que me apoiaram e firmaram minha base para conquistar os meus objetivos ao longo desses quatro anos dentro do IFRN.

Aos meus colegas de classe, que me acompanharam e apoiaram durante todo o período que passamos juntos no instituto.

Aos meus amigos, Liandra Kaylane, Karydja França, Kauã Carvalho, Gelsifran Santos, Jhenyffer Floripes, Lucas Felipe, Clara Pinheiro, Ana Luiza Ferreira, Kaylane Freire, Eduarda Chaves e a todas as pessoas que me ajudaram, de forma direta ou indireta, a seguir os meus objetivos e sonhos.

Aos meus orientadores, Elenilson Vieira, Francisco Magno e Irlan Arley, que me orientaram paciente, profissional e amigavelmente, auxiliando-me, no possível, para que este trabalho fosse finalizado na sua excelência.

À minha parceira de TCC, Clara Yasmin, que lindamente me acompanhou, com paciência e responsabilidade, ajudando-me e enxugando-me as lágrimas, estando do meu lado ao longo desses dois anos de duração deste trabalho.

Ao IFRN, instituição essa que me formou acadêmica e politicamente, à qual sou muito grato por todas as experiências vivenciadas, sejam elas boas ou ruins.

Por fim, àqueles que eu não ouso chamar apenas de amigos: Pedro Hugo, José Daniel, Yasmin Carvalho, Caio Rian, Antonio Marcos e Esaú Rodrigues. Tais nomes representam uma irmandade que formou minha estrutura ao longo desses anos, e, principalmente, durante a pandemia. Foram eles que me fizeram não desistir e me salvaram de diversas formas quanto eu mais precisei. Muito obrigado!

"O estilo é o próprio homem".
Conde de Buffon

RESUMO

A atribuição de autoria consiste no estudo de uma produção para determinar seu autor. Embora seja uma prática antiga, que remonta a séculos, a importância e necessidade da atribuição de autoria não diminuíram com o tempo, ao contrário disso: conforme as novas demandas nessa área, os métodos usados para sua realização seguiram os próprios avanços tecnológicos. Nesse contexto, surgiram inúmeros softwares que ajudam ou atribuem completamente a autoria de uma obra a um autor, levando em conta as mais variadas características da escrita individual. Tais características, inerentes aos autores, são chamadas de "impressão digital autoral". Como não há consenso na comunidade científica sobre qual é a característica mais eficiente para essa diferenciação, a escolhida para este projeto foi a sequência de classificações gramaticais. Assim, este trabalho apresenta o desenvolvimento de um software que atribui a autoria de textos levando em consideração a característica supracitada. O algoritmo de compressão de dados, PPM-C, foi o método selecionado para o desenvolvimento do software apresentado neste trabalho e mostrou-se uma escolha acertada graças aos bons resultados. Em seguida, foram feitas implementações baseadas em trabalhos semelhantes, visando melhorar a assertividade do programa, além de diversos testes para definir as melhores condições de funcionamento do sistema. Por fim, os resultados foram favoráveis, uma média de 86,5% de acertos nas melhores condições do sistema, especialmente se considerados os desafios de trabalhar com duas ciências tão distintas como ciência da computação e linguística.

Palavras-chaves: Atribuição. Autoria. PPM-C. Literatura brasileira. Estilo.

ABSTRACT

The attribution of authorship consists in the study of a production to determine the author. Although it is an old practice, dating back centuries, the importance and necessity of attributing authorship has not diminished over time, on the contrary: according to the new demands in this area, the methods used to carry it out followed the technological advances themselves. In this context, numerous software emerged that help or completely attribute the authorship of a work to an author, taking into account the most varied characteristics of individual writing. Such characteristics, inherent to authors, are called "authorial fingerprinting". As there is no consensus in the scientific community about which is the most efficient feature for this differentiation, the one chosen for this project was the sequence of grammatical classifications. Thus, this work presents the development of a software that attributes the authorship of texts taking into account the aforementioned characteristic. The data compression algorithm, PPM-C, was the method selected and it proved to be the right choice thanks to the good results. Afterward, implementations were made based on similar works, aiming to improve the program's assertiveness, in addition to several tests to define the best operating conditions for the system. In conclusion, the results were favorable, an average of 86.5% of correct answers in the best conditions of the system, especially considering the challenges of working with two sciences as distinct as computer science and linguistics.

Keywords: Attribution. Authorship. PPM-C. Brazilian Literature. Style.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABPI	*Associação Brasileira da Propriedade Intelectual*
API	*Application Programming Interface*
AWS	*Amazon Web Service*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protocol*
IDE	*Integrated Development Environment*
JVM	*Java Virtual Machine*
JPA	*Java Persistence API*
JPQL	*Java Persistence Query Language*
JSON	*JavaScript Object Notation*
KLD	*Kullback-Leibler Divergence*
KNN	*K-Nearest Neighbor*
LDA	*Linear Discriminant Analyses*
LOO-CV	*Leave-One-Out Cross-Validation*
MDA	*Multiple Discriminant Analysis*
MIME	*Multipurpose Internet Mail Extensions*
MVC	*Model-view-controller*
NSC	*Nearest Shrunken Centroids*
PCA	*Principal Components Analysis*
PPM	*Prediction by Partial Matching*
PPM-C	*Prediction by Partial Matching - C*
RDA	*Regularized Discriminant Analysis*
SGBD	*Data Base Management System*
SSP	*Simple Started Project*

STS	*Spring Tools Suite*
SVM	*Support Vector Machine*
S3	*Simple Storage Service*
UI	*User Interfae*
URI	*Uniform Resource Identifier*
USP	*Universidade de São Paulo*
UX	*User Experience*
VSCODE	*Visual Studio Code*

LISTA DE SÍMBOLOS

σ	Letra grega Sigma

SUMÁRIO

1 INTRODUÇÃO

Um dos temas mais debatidos, hodiernamente, é a questão da autoria, ensejando reflexões no âmbito acadêmico, jornalístico e mesmo jurídico a partir, sobretudo, do grande aumento de casos de plágio e *Fake News*. No entanto, essa perspectiva problemática da autoria resulta de uma evolução histórica do próprio termo, hoje identificado diretamente com os conceitos de "originalidade" e de "propriedade", por exemplo, no campo das produções acadêmicas.

No século XVII, particularmente no que diz respeito à literatura, os escritores se importavam mais em emular padrões retóricos e poéticos das "autoridades" – isto é, dos autores de prestígio – do seu tempo e da cultura greco-latina, exercitando e aperfeiçoando códigos de linguagem reconhecidos e partilhados na sociedade letrada em que viviam. É o caso do poeta barroco Gregório de Matos (1636-1696), cuja "autoria" dos textos que lhe são atribuídos mesura-se por uma semelhança de traços linguísticos/poéticos e não necessariamente por uma atribuição biográfica, subjetiva, "assinada". Gregório, como muitos poetas barrocos, foi um "imitador" de outros poetas com os quais partilhava semelhanças retórico-poéticas a partir do exercício da imitação de temas, figuras de linguagem e construção sintática.

De acordo com João Adolfo Hansen (HANSEN, 2004), era comum que as obras seguissem padrões e que os autores até mesmo abdicassem de suas autorias para incluí-las no repertório de nomes mais famosos, o que mais tarde ocorreu com o próprio Gregório de Matos: "[...] o Licenciado unifica em códice tantas obras de gênero e formas diversos, conferindo a sua autoria à unidade do nome próprio, 'Gregório de Matos e Guerra', porque, como um letrado do século XVIII, constitui uma tradição local" (HANSEN, 2004). Portanto, o descrito "Licenciado" não fez nada que outro escritor de sua época também não fizesse, como explica o referido estudioso, incluindo seu trabalho na coletânea de "uma etiqueta ou um dispositivo discursivo, unidade imaginária e cambiante nos discursos que o compõem" (HANSEN, 2004), nesse caso, Gregório de Matos.

A "liberdade poética" que marcou as produções literárias barrocas, porém, mudou com o advento do Romantismo, a partir do século XVIII. Assim, a perspectiva de autoria coletiva deu lugar à ideia de subjetividade estilística, isto é, o estilo assinado de cada autor, que passou a esmerar-se a fim de distinguir-se dos outros, buscando manter seu próprio conjunto de traços estilísticos, não obstante partilhando temas e características poéticas e filosóficas da mesma escola literária. Desse modo, na literatura brasileira do séc. XIX, embora haja semelhanças entre Casimiro de Abreu (1839-1860), Gonçalves Dias (1823-1864), Fagundes Varela (1841-1875) e Castro Alves (1847-1871), por exemplo, há igualmente diferenças que permitem distingui-los enquanto singularidades estilísticas

– sem mencionar a grande subjetividade na poesia romântica brasileira, que é o poeta Sousândrade (1833-1902)(PINTO, 2017).

Por conseguinte, as transformações quanto à compreensão de autoria, no contexto do séc. XIX, não se restringiram aos literatos, ganhando espaço em outras áreas do conhecimento. Em 1851, o matemático e lógico britânico Augustus De Morgan, usando os métodos que conhecia, tentou mensurar vocábulos para atribuir autoria dos textos por ele estudados.

Assim, é a partir do Romantismo – e de todo o contexto libertário que o antecedeu no âmbito filosófico, político, social e econômico – que o significado de autoria evolui para o sentido que atualmente lhe é dado, grosso modo, como sinônimo dos referidos conceitos de "originalidade" e de "propriedade". Esse novo entendimento do termo, por seu turno, passou a nortear diversas atividades no campo acadêmico, técnico-científico e jornalístico da atualidade e seus respectivos impasses, nomeadamente a atribuição de propriedade intelectual nas produções acadêmicas, combatendo o plágio; e a testagem da origem e veracidade das informações, coagindo a multiplicação de informações falsas, hoje vulgarizadas como *Fake News*.

No particular às produções textuais, essa nova perspectiva de autoria mesura-se justamente a partir da concepção de "autor" enquanto individualidade que, por sua vez, é dona da sua "obra", cuja propriedade autoral deve estar resguardada de plágio – entendido este não na dimensão de prática criativa, como variação, e sim enquanto apropriação indevida da propriedade alheia, isto é, crime contra os direitos autorais. Quanto a esse problema, o diretor relator da Associação Brasileira da Propriedade Intelectual (ABPI), Benny Spiewak, alertou para o caráter duplo em que consiste a Internet, já que, mesmo sendo um suporte público de grande alcance para a propalação de pesquisas, serve como um ampliador para práticas ilegais como o plágio (SANTIAGO, 2018).

A prova da coexistência dessas realidades – aumento da produção intelectual e de práticas que ameaçam sua segurança – é o recente caso na universidade norte-americana de Harvard: dados institucionais, divulgados pelo site Gazeta do Povo, relatam que, em média, 17 alunos são afastados anualmente da universidade por conduta inadequada no que se refere a essas questões (SANTIAGO, 2018), não obstante a página oficial da instituição seja bastante clara sobre a política e padrão de integridade acadêmica exigidos (HARVARD, c2019).

Paralelamente ao crescimento do plágio, também é perceptível o aumento substancial de *Fake News*. Apesar dos problemas delas resultantes, as conhecidas "notícias falsas" ainda não podem ser consideradas crime no Brasil, uma vez que não estão assim previstas legalmente (e, segundo o inciso XXXIX do 5º artigo da Constituição Federal de 1988, um ato é infracional apenas se houver uma lei que, previamente, defina-o como tal) (BRASIL, 1988). Apesar dos trâmites de projetos para a criminalização dessa prática no Congresso

Nacional, nenhuma medida concreta foi tomada (AMARAL, 2020). Quase que exclusivas dos meios digitais, essas pseudonotícias são muito comuns no Brasil, levando o país a ocupar o terceiro lugar no ranking de países que mais sofrem com o aumento de *Fake News*, de acordo com o *Reuters Institute Digital News Report* em 2018 (FORBES, 2018).

Tanto o plágio quanto as *Fake News* são dois problemas que se agravaram com o avanço da tecnologia da informação, a qual facilitou exponencialmente o acesso a uma diversidade de conteúdos, a exemplo da produção acadêmica de autores de várias regiões, línguas e épocas, bem como a uma profusão de notícias e informações que, muitas vezes, não correspondem à realidade. São, portanto, dois problemas que prejudicam e desestabilizam áreas estratégicas, como a produção acadêmica, as instituições políticas, a segurança e o bem-estar social etc.

Em face dos dois referidos problemas – diretamente relacionados, conforme exposto, com a perspectiva atual de autoria e as novas dinâmicas e usos da tecnologia da informação –, evidenciou-se, primeiramente, a necessidade de pensar alternativas para mitigá-los por meio da verificação de origem das informações. Por conseguinte, surgiu a seguinte dúvida: de que maneira seria possível, então, realizar essa verificação de forma mais prática e eficaz?

Em resposta a essa pergunta, este trabalho busca desenvolver um software destinado à atribuição de autoria, a ser utilizado, inicialmente, na identificação autoral de textos literários brasileiros.

Na perspectiva de operacionalização desse objetivo, o presente trabalho propõe: revisar pesquisas relacionadas especificamente com a área da tecnologia da informação, assim como relativos aos conceitos de estilística e gramática que estejam ligados ao tema principal deste projeto; organizar, a partir de textos disponibilizados pelo usuário, um corpus padronizado automaticamente nos moldes do sistema; codificar e comprimir, através do algoritmo PPM-C, o corpus previamente processado, gerando um conjunto de arquivos que permitam, estatisticamente, a confirmação ou refutação das suas respectivas autorias; definir as condições de melhor funcionamento do sistema, por meio da apresentação de diferentes resultados (levando em conta variações de parâmetros como o tamanho dos textos de entrada e diferentes contextos); detalhar modificações e aprimoramentos feitos no sistema; e, por fim, criar um serviço Web e um site (protótipo) onde o método possa ser disponibilizado para o público.

Sob essa ótica, parte-se da hipótese de utilidade do software desenvolvido e aqui apresentado – intitulado WhoAuthor – na verificação autoral, o qual, por meio de dados prévios de possíveis autores, realiza análises precisas e, com facilidade e rapidez, confirma ou nega supostas autorias.

Assim, no primeiro capítulo, é apresentado, a partir de textos disponibilizados pelo

usuário, o processo de organização de um *corpus* padronizado nos moldes do sistema de forma automática.

Em seguida, no segundo capítulo, explica-se a codificação e compressão, através do algoritmo PPM-C, do *corpus* previamente processado, gerando um conjunto de arquivos que permite, estatisticamente, a confirmação ou refutação das suas respectivas autorias.

Na sequência, é descrita a definição das condições de melhor funcionamento do sistema, levando em conta diferentes resultados, que divergem entre si em variações de parâmetros como o tamanho dos textos de entrada e contextos, objetivando, com isso, a melhor margem de acerto possível.

Finalmente, são abordadas as diversas modificações feitas no código, ao longo de todo o processo de desenvolvimento da pesquisa, com o fito de agilizar o processo de treinamento, melhorar os resultados, utilizar plataformas mais atuais etc. Nesse subcapítulo, discorrer-se-á, também, sobre a confecção de um site, criado para facilitar a interação com o usuário e divulgar o sistema.

Ao término do presente trabalho, evidencia-se que os objetivos e a hipótese ensejados pelo mencionado problema foram atendidos, confirmando a viabilidade do software desenvolvido e aqui apresentado para a atribuição, inicialmente, de autoria de textos literários, podendo estender-se, também, para outros objetivos afins, a exemplo da verificação de notícias no combate às *Fake News*.

1.1 OBJETIVO GERAL

Desenvolver um software para atribuir autoria de textos da literatura brasileira através do reconhecimento de padrões de sequências de classificações gramaticais com a utilização do PPM-C.

1.2 OBJETIVOS ESPECÍFICOS

- Investigar trabalhos relacionados e estudar conceitos de linguística que estão relacionados ao tema;
- Pesquisar livros literários em bases públicas para a extração de textos;
- Definir estrutura do *corpus* para a realização dos treinamentos e testes;
- Codificar e comprimir o *corpus*, através do PPM-C, para gerar estatísticas que permitam a atribuição de autoria;
- Realizar testes para ajustar os parâmetros a fim de melhorar os resultados de atribuição de autoria;

- Detalhar as modificações e melhorias feitas no sistema em relação ao protótipo inicial DWE;
- Criar uma aplicação *full-stack* completa, com acesso a banco de dados e serviço web consumido por um protótipo de *front-end* que permita a utilização da ferramenta;
- Submeter um artigo para um congresso ou periódico especializado;
- Realizar o registro oficial do software.

2 FUNDAMENTAÇÃO TEÓRICA

O referencial teórico que corresponde à base deste trabalho engloba problemáticas atuais, as quais fornecem subsídio para a relevância do WhoAuthor; conceitos da estilística e sua relação com a área de informática; uma série de estudos que lapidaram características fundamentais do sistema; além de uma breve explanação sobre o algoritmo de compressão PPM-C. Todos esses pontos serão abordados em seguida, como forma de tornar compreensíveis os direcionamentos tomados.

2.1 AUTORIA E PLÁGIO

Embora o avanço da tecnologia da informação tenha facilitado o acesso a dados (a exemplo de pesquisas e citações da produção textual de uma infinidade de autores de várias regiões, línguas e épocas), a devida origem dessas informações, todavia, nem sempre é identificada consoante orientações técnicas, éticas ou mesmo jurídicas, resultando no problema de atribuição equivocada de autoria que envolve dois conceitos interligados: a própria concepção de *autoria* e o conceito negativo de *plágio*.

Cumpre observar, desde logo, que a concepção atual de autoria resulta de transformações ocorridas, sobretudo, nos séculos XVIII e XIX, no contexto pós-iluminista e romântico, a exemplo da própria literatura romântica. Diferentemente de épocas anteriores, como o Barroco, onde a autoria tinha sua significância e produtividade através da unificação (HANSEN, 2004), o Romantismo brasileiro buscou diferenciar-se de tendências vigentes e definir um modelo próprio, sem a influência de inspirações externas.

O movimento romântico intencionava, assim, estabelecer um modelo próprio de produção escrita, dando enfoque a elementos nacionais, constituindo-se como movimento pioneiro na visão crítica que considera o ser humano de forma original e singular, naturalmente provido de gênio e liberdade (RONCARI, 1995). Exemplo dessa "genialidade", nas letras brasileiras, é o caso do poeta Joaquim de Sousa Andrade (1833-1902), o Sousândrade, cuja linguagem rica e diferencial ia de encontro ao que era considerado canônico na época, e isso, ao passo que ajudava a ressignificar a pertinência da autoria, contribuindo para a ideia de subjetividade, desagradava a um considerável público (PINTO, 2017).

Nesse contexto, os escritores passaram a tornar mais explícita a ligação pessoal com seus trabalhos e optar pelas suas próprias individualidades em vez dos parâmetros pré-estabelecidos de épocas passadas: o conceito de autoria evoluiu, então, conforme a entendemos hoje, isto é, diretamente relacionada à ideia subjetiva de "originalidade" e de "propriedade".

Em nosso tempo, a questão de autoria mesura-se, portanto, em função dessa

concepção de “autor” enquanto individualidade que é proprietária da sua “obra”. Tal perspectiva incide, inclusive, sobre aspectos legais que, por sua vez, buscam resguardar a propriedade autoral e, por conseguinte, coibir a prática do plágio. É a partir dessa perspectiva de individualidade que os próprios vocábulos são conceituados: a *autoria* como “qualidade ou condição de autor” e o *plágio* como “imitação de trabalho, geralmente intelectual, produzido por outrem” [1] (MICHAELIS,).

Logo, o plágio é tido aqui como problema a ser resolvido, em razão do constante aumento da sua prática, por exemplo, nos meios acadêmicos. No intuito de combatê-lo, muitas universidades, incluindo a Universidade de São Paulo (USP), têm a tecnologia como aliada (SANTIAGO, 2018). Nesse cenário, programas de computadores capazes de apontar nos textos similaridades indicativas de plágio, progressivamente, tomam espaço nos estudos para garantir a legitimidade da autoria. Tais circunstâncias justificam e reafirmam a relevância do desenvolvimento e aprimoramento de programas como o WhoAuthor, aqui apresentado.

2.2 A GRAMÁTICA E A ESTILÍSTICA

Em face dos problemas decorrentes do aumento dos casos de plágio, nomeadamente na esfera acadêmica, chegou-se à conclusão de que a melhor maneira de combatê-los seria por meio da verificação autoral.

Nessa perspectiva, pensou-se em desenvolver um software destinado, inicialmente, à atribuição de autoria em textos literários. Assim, considerando a natureza literária dos autores utilizados – tanto os que formam o *corpus* que corresponde aos dados de treinamento, quanto os autores dos textos de entrada –, fez-se necessário, por conseguinte, analisar a estatística de dados gramaticais, sem desconsiderar, entretanto, a perspectiva estilística dos textos em questão.

De acordo com Ruy Araújo (ARAUJO, s.d.), a diferença entre gramática e estilística consiste em que, enquanto a primeira se limita ao estudo das formas linguísticas para o estabelecimento da compreensão na comunicação, a segunda explora a expressividade da linguagem e sua capacidade de emocionar e sugestionar o ser humano/leitor. Dessa maneira, segundo o referido professor, enquanto a gramática analisa a linguagem intelectiva, a estilística aborda a linguagem afetiva. Ruy Araújo (ARAUJO, s.d.) evidencia essa ideia ao defender que o estilo é a forma de expressão individual de cada pessoa, e isso ocorre por intermédio de uma coleção de procedimentos que exprimem, na representatividade da língua, traços psicológicos do escritor.

Essa concepção de estilística coincide com o próprio conceito de estilo abordado neste trabalho, razão pela qual a compreensão dessas definições é imprescindível para

[1] Essa dimensão negativa do plágio não considera, obviamente, imitações com propósito explícito de prática criativa, a exemplo da paródia, do pastiche etc.

o desenvolvimento do sistema de atribuição automática de autoria aqui apresentado. Consoante Buffon (1707-1788), "o estilo é o homem" – o que significa dizer que o conceito de estilo é fundido à personalidade humana individual (CHOCIAY, 1983).

Ao contrário de textos de caráter puramente informativo/denotativo, que já são subjetivos o suficiente para um algoritmo, o grau de subjetividade de um texto literário pode ser alto o bastante para dificultar a sua compressão. Todavia, apesar de serem grandes as limitações para a consideração desses aspectos literários no domínio da informática, como uma ciência exata, eles têm de ser levados em conta, pois a conceituação de *impressão digital autoral* equivale a um conjunto de características inerentes a um autor. Essa impressão digital está estritamente ligada à estilística, principalmente se compararmos esse termo com o conceito de estilo apresentado por Ruy Araújo (ARAUJO, s.d.). Ainda de acordo com ele, os traços estilísticos representam a marca de cada autor, a junção de toda sua produção sob uma ótica de ideais estéticos que se projetam na língua.

Dessa forma, sendo a impressão digital autoral um conjunto de características próprias de um autor, muitas vezes aplicadas de forma inconsciente em um texto e inerentes a quem o produziu, é possível afirmar que a estilística e a impressão digital autoral são conceitos que se assemelham. Tal semelhança indica que a referida nomenclatura – hoje utilizada por profissionais de informática para definir a forma de escrever de um autor – equivaleria, grosso modo, ao antigo conceito de "estilo" da linguística.

As áreas fônica, sintática, morfológica e semântica – todas consideradas pela linguística – também são tratadas como características constituintes de uma impressão digital autoral: o uso de onomatopeias; pontuação como reticências para indicar interrupção ou velação de um pensamento; e até a presença de palavras funcionais[2] podem ser explicados e inseridos nas áreas abordadas pela estilística.

A partir dessas considerações sobre os conceitos de gramática e estilística, o sistema aqui apresentado foi desenvolvido e continua em constante processo de aperfeiçoamento, levando em conta não só a intelectividade da linguagem, como também a sua afetividade, visando uma análise cada vez mais abrangente de textos literários, a fim de garantir ao usuário do programa uma atribuição de autoria automática efetiva e segura. Os resultados obtidos já são uma prova da capacidade do método apresentado, dadas todas as implicações que impedem a perfeição.

2.3 PPM-C

O PPM, abreviação para *Prediction by Partial Matching*, é um recurso para compressão de dados que possui um codificador e um modelo estatístico dos informes,

[2] Aquelas que são pobres em conteúdo semântico próprio, de certa forma irrelevantes para a recuperação de informações e que independem de tópicos, sendo muito utilizadas por um autor para a expressão de conceitos, como preposições, artigos e conjunções.

trabalhando com compressão de dados. Seu diferencial está no fato de atribuir probabilidade por um contexto e não só pela frequência absoluta. Assim, um codificador aritmético adaptativo receberá no *input* um símbolo **S**, e o codificará com probabilidade **P**. Pelo fato de utilizar um conjunto de **K** símbolos antecessores, é possível atribuir a probabilidade de **S'** dos símbolos que ocorrem em seguida de **S** (CLEARY; TEAHAN; WITTEN, 2003).

Como foi explicado por Barufaldi (BARUFALDI et al., 2010), dado um novo símbolo **S** a ser comprimido em um contexto C_k de tamanho **K** (neste projeto foram utilizados tamanhos de contexto para dois a nove símbolos), o PPM utiliza seu modelo estatístico para calcular a probabilidade condicional da ocorrência do símbolo **S** e passa essa probabilidade para o codificador aritmético. Caso não haja ocorrência do símbolo **S** no contexto C_k, um símbolo especial de **ESCAPE** é codificado e é realizada uma nova busca no contexto C_{k-1}, que é a sequência de símbolos C_k reduzida de um símbolo. Caso o símbolo não seja encontrado em nenhum dos contextos, ele é codificado utilizando um modelo que considera equiprováveis todos os símbolos possíveis de ocorrer. Após a codificação do símbolo, o modelo atualiza as probabilidades condicionais do símbolo **S**. Esse processo é repetido para cada novo símbolo a ser comprimido.

É atribuída pelo codificador uma quantidade de bits inversamente proporcional à probabilidade que este possui. Iniciando com intervalos de 0 a 1, a probabilidade de cada símbolo origina um novo intervalo. Para igualar a entropia da fonte a ligação entre símbolos particulares e palavras-código é extinta. Esse algoritmo representa a probabilidade de ocorrência dos símbolos seguindo estes intervalos. Os intervalos são subdivididos proporcionalmente pela probabilidade da tabela de intervalos, isso para cada símbolo seguinte. Assim, é descoberto um intervalo referente à possibilidade de ocorrência de cada elemento (CLEARY; TEAHAN; WITTEN, 2003). Uma tabela é construída pelo modelo estatístico das probabilidades, seguindo esta fórmula:

$$P(\sigma) = \frac{n_\sigma}{N} \tag{2.1}$$

$\mathbf{P}(\sigma)$ representa a probabilidade do símbolo σ ocorrer, n_σ representa quantas vezes esse símbolo apareceu e **N** se limita à dimensão do arquivo.

2.4 TRABALHOS RELACIONADOS

Nesta seção, apresentar-se-á para fins informativos, comparativos e como meio de justificar a positividade das decisões tomadas uma revisão de diversos estudos que, em suas particularidades, contribuíram para o desenvolvimento do presente trabalho. Todos eles envolvem a atribuição de autoria automática, diferencindo-se quanto aos métodos utilizados, os textos usados como dados e outras características citadas a posteriori.

A pesquisa sobre essa diversidade de métodos de atribuição de autoria que vem sendo utilizados na comunidade científica foi bastante útil para assegurar a escolha do PPM-C e de todos os outros aspectos do WhoAuthor.

Ebrahimpour (EBRAHIMPOUR et al., 2013) obteve mais de 90% em acertos no desenvolvimento de um software para atribuir autoria por meio da Máquina de Suporte de Vetores (SVM) e da Análise Discriminante Múltipla (MDA): o primeiro é um algoritmo que distribui os dados pela definição de um hiperplano, sendo popular quando se trata de categorizar informações; o segundo, por sua vez, pode ser descrito como um recurso de análise que concede casos conhecidos pela discriminação de variáveis preditoras. Os autores ainda utilizaram a validação cruzada de exclusão única (LOO-CV) para avaliar a performance dos dois métodos, ferramenta essa que faz testes com textos retirados dos dados de treinamento e obtém a precisão da classificação pela divisão do número de textos classificados corretamente pelo número total de textos.

Utilizando o software para questões lexicométricas, chamado Lexico3, Brandão (BRANDAO, 2006) fez um estudo sobre a autoria das "Cartas Chilenas". Mesmo com alguns entraves, o programa recebe uma boa avaliação do autor, sendo caracterizado como poderoso e facilmente manuseável. Vale notar que essa aplicação não realiza a atribuição de autoria, atuando apenas como auxiliar, pois o trabalho é realizado por um humano, diferentemente de outras pesquisas aqui abordadas, como a de Barufaldi (BARUFALDI et al., 2010). Esta última, usando o PPM-C (uma variante do PPM), apresentou a criação de um sistema que automaticamente indicava, a respeito de uma obra literária brasileira, o período literário em que ela foi produzida. O PPM foi usado por ser efetivamente bom na classificação de textos. A particularidade desta variante (C) está em seu mecanismo que exclui símbolos não ocorrentes ao iniciar a codificação, melhorando a compressão (CLEARY; TEAHAN; WITTEN, 2003).

No extenso trabalho de Juola (JUOLA, 2006), por conseguinte, são abordadas as mais diversas características que podem servir como uma "impressão digital autoral", bem como métodos utilizados para atribuir autoria. Após seus estudos, ele conclui que não existe um consenso entre os autores sobre a melhor característica para identificar a autoria de um trabalho. Ao falar dos métodos utilizados, o SVM é colocado como o que fornecia os melhores resultados, até aquele momento; no entanto, o autor afirma que, futuramente, ele será superado por métodos mais eficientes. Isso se materializa no trabalho de Ebrahimpour (EBRAHIMPOUR et al., 2013), onde o MDA obtém números melhores que o SVM, na taxa de acerto.

A obra de Juola (JUOLA, 2006), assim como a de Brandão (BRANDAO, 2006), que apresenta a atribuição de autoria e seu histórico, foram essenciais para a construção de um repertório histórico e científico sobre a atividade da atribuição de autoria. Conhecer a história dessa atividade é muito importante para compreender seu estado atual, tal

como suas principais necessidades. Apesar de eficiente, o software utilizado por Brandão (BRANDAO, 2006) apenas ajuda profissionais a atribuir autoria – ao contrário do programa apresentado neste trabalho, que apesar de não ter 100% de precisão, facilmente pode ser usado por qualquer pessoa e realiza todo o processo de atribuição.

Ainda quanto às altas taxas de acerto obtidas no referido trabalho de Ebrahimpour (EBRAHIMPOUR et al., 2013), elas podem ser justificadas, em parte, pelo número bastante superior de textos por autor, em relação ao programa aqui apresentado. Enquanto ele precisou de até 14 textos por autor, o presente trabalho contou com somente 4 amostras de 4 obras diferentes para cada autor. Uma grande quantidade de obras pode melhorar a taxa de acerto, mas infelizmente essa situação não pode ser aplicada a casos muito ocorrentes, onde o autor possui poucos textos disponíveis para treinamento do sistema. Assim, este trabalho apresenta vantagens, já que obteve altas taxas de acerto com uma quantidade razoavelmente baixa de dados de treinamento, simulando situações reais.

Em seu trabalho sobre atribuição de autoria, Stamatatos (STAMATATOS, 2017) propõe um método de distorção de texto onde a estrutura textual que se refere ao estilo pessoal do autor é mantida e palavras menos frequentes, normalmente relacionadas à temática, são mascaradas, com o objetivo de melhorar a eficiência da atribuição. Nesse princípio, sustenta-se o trabalho do autor. Apesar das melhorias que o método propõe, o próprio Stamatatos (STAMATATOS, 2017) reconhece que é uma tarefa difícil e bastante subjetiva definir até que ponto se estende o estilo do autor, uma vez que não é esperado que estruturas textuais pessoais sejam modificadas por variação no gênero ou tópico. Contudo, no campo da linguagem, esse fato nem sempre pode ocorrer como uma regra, conforme anteriormente abordado no tópico sobre as relações entre gramática e estilística.

Apesar do presente trabalho não adotar o mesmo método de atribuição que Stamatatos, existem concordâncias em questões como a pontuação. Em ambos os trabalhos, ela foi tida como um fator relevante para atribuir autoria, seguindo o apontamento de Stamatatos (STAMATATOS, 2017), de que sinais de pontuação e símbolos são marcadores importantes.

Além da questão da pontuação, o WhoAuthor também se assemelha, em outros aspectos, ao que escreve Stamatatos (STAMATATOS, 2017), a exemplo da utilização de conjunto fechado e da análise morfológica, pontos estes dentre os que norteiam o presente trabalho. Apesar da limitação de conteúdo, conjuntos fechados são amplamente utilizados nessa área de pesquisa pela segurança que oferecem ao pesquisador na contabilização dos resultados; ademais, a análise morfológica – incorporada no WhoAuthor por meio das sequências de classificações gramaticais – mostra-se mais eficaz e menos ruidosa quando comparada à sintática e semântica (STAMATATOS, 2017) .

Por seu turno, Zhao e Zobel (ZHAO; ZOBEL, 2007), em seu trabalho sobre atribuição de autoria, estilo e literatura clássica, objetivam fazer uma comparação da

proficiência de métodos de atribuição de autoria, a fim de examinar a eficácia dessa atividade na literatura. Para isso, é feita uma análise de diferentes métodos de atribuição, incluindo alguns como o SVM, aqui apresentado, e o Kullback-Leibler Divergence (KLD), proposto pelos autores em trabalhos anteriores, sendo este último uma medida de entropia relativa que apresenta resultados equiparáveis ao SVM, método de prestígio na área de atribuição (ZHAO; ZOBEL, 2007).

Zhao e Zobel (ZHAO; ZOBEL, 2007) também pontuam a popularidade e efetividade de marcadores como palavras funcionais, classes gramaticais e sinais de pontuação em estudos de atribuição de autoria, o que mostra, mais uma vez, a assertividade das escolhas feitas para o WhoAuthor. Mesmo que Zhao e Zobel (ZHAO; ZOBEL, 2007) definam esses marcadores como limitados, os resultados dessa pesquisa demonstram que isso não foi um aspecto negativo, uma vez que foram obtidas porcentagens de acerto favoráveis. Portanto, conclui-se que os autores possuem estilos próprios e identificáveis e que os marcadores, por sua vez, têm eficiência para detectar e auxiliar a classificação, até mesmo os mais simples.

Em consonância com a referida argumentação de João Adolfo Hansen (HANSEN, 2004), Zhao e Zobel (ZHAO; ZOBEL, 2007) discorrem a respeito das dificuldades em classificar textos de determinados autores, pelo fato de o conceito de autoria ter sido moldado no decorrer dos anos. Assim, os respectivos exemplos da preocupação de escritores como Gregório de Matos (HANSEN, 2004) e Oscar Wilde (ZHAO; ZOBEL, 2007) em adequar suas escritas ao padrão da época sufocou alguns traços de suas impressões digitais autorais, complicando significativamente o trabalho de atribuidores de autoria automáticos.

Nessa perspectiva, Baayen (BAAYEN et al., 2002) e outros pesquisadores apresentam um experimento onde analisam a atribuição de autoria automática controlando variáveis, em relação aos autores e seus respectivos textos, que poderiam camuflar a real eficácia do método em questão. Os pesquisadores recrutaram estudantes universitários, escritores não profissionais, com históricos parecidos e, então, oferecendo-lhes o mesmo incentivo, direcionamento de tópico e gênero, solicitaram a produção de alguns textos. Tais escritos foram o material estudado por meio da *principal components analysis* (PCA) e da *linear discriminant analyses* (LDA), mostrando-se o segundo método mais eficiente e, dessa maneira, respondendo ao questionamento dos pesquisadores.

A dubiedade fundamental do trabalho era sobre a existência da impressão digital autoral, já que muitos estudos – incluindo este – partem do pressuposto de que ela existe. Assim, os experimentos comprovaram a existência desse estilo pessoal, apesar de ressaltarem que, em alguns casos, a sua identificação pode ser mais difícil, como em textos de imposição editorial (BAAYEN et al., 2002). As pesquisas de Baayen (BAAYEN et al., 2002) constatam, também, que o enriquecimento da matriz de dados com a contabilização dos sinais de pontuação aperfeiçoa o processo de atribuição e – à semelhança deste trabalho – realiza seu estudo por meio de um experimento controlado. Evidentemente, pelo fato de

terem tido contato com os autores, Baayen (BAAYEN et al., 2002) conseguiram exercer um controle maior sobre os dados que seriam analisados, uma tarefa quase impossível no caso do WhoAuthor, uma vez que foram escolhidos literatos brasileiros, muitos já falecidos e com obras publicadas por diferentes editoras e em diferentes épocas.

Jockers e Witten (JOCKERS; WITTEN, 2010), por conseguinte, desenvolvem a sua pesquisa a partir da constatação da carência de estudos que fizessem comparação entre os métodos de atribuição de autoria e a falta de consenso entre os estudiosos sobre qual é o melhor método. Os cientistas fizeram um estudo comparativo de aprendizado de máquina analisando 5 métodos: o Delta, *k-nearest neighbors* (KNN), o SVM, o *nearest shrunken centroids* (NSC) e a *regularized discriminant analysis* (RDA). Ponderando os prós e contras de cada um dos métodos, os que apresentaram melhor desempenho, conforme Jockers e Witten (2010), foram o NSC e a RDA. Em concordância com a bibliografia aqui apresentada, novamente é defendida a relevância de palavras funcionais (JOCKERS; WITTEN, 2010) . A escolha pelo uso de poucos textos para treinamento dos autores, por sua vez, também é assegurada por Jockers e Witten (JOCKERS; WITTEN, 2010), tornando o WhoAuthor mais viável para o uso.

A par dessas considerações, pode ser percebida a aproximação do WhoAuthor com trabalhos bem-sucedidos. O projeto de Barufaldi (BARUFALDI et al., 2010), por exemplo, serviu como fonte de inspiração para este trabalho. A partír da ideia de trabalhar com literatura brasileira, por ele adotada, foi tomado outro rumo ao ser decidido que o presente trabalho iria focar na identificação de autores e não de períodos literários, como os pesquisadores do referido projeto realizaram. Seus passos também foram seguidos, neste projeto, no que se refere à escolha da variante C do PPM. Logo, é possível observar que os princípios que norteiam o WhoAuthor estão em concordância com a comunidade científica, a saber: a consideração da existência e relevância de uma "impressão digital autoral" no processo de atribuição de autoria (EBRAHIMPOUR et al., 2013), a escolha de um marcador eficiente como as sequências de classificações gramaticais (ZHAO; ZOBEL, 2007), a importância dada aos sinais de pontuação (BAAYEN et al., 2002) e até a quantidade de textos utilizada (JOCKERS; WITTEN, 2010). Para além dessas concepções, os resultados dos trabalhos relacionados, expressos numericamente, reafirmam a sua paridade com o WhoAuthor. Apesar da comparação de números não ser a forma mais ideal de contrastar o WhoAutor de outros sistemas - uma vez que os demais pesquisadores utilizaram algoritmos diferentes do PPM-C, estruturaram os testes de seus estudos de maneira distinta do que foi realizado neste trabalho e até os cálculos para precisar a assertividade foram díspares -, ele se equipara aos resultados apresentados por Baayen e seus colegas (BAAYEN et al., 2002), que nas melhores hipóteses, ficaram entre 81,5% e 88,1%, e também a métodos como o SVM e MDA, que apresentaram desempenho superior a 90% (EBRAHIMPOUR et al., 2013). Entretanto, esses números só foram possíveis, nos referidos experimentos, devido à grande quantidade de textos utilizados, enquanto no WhoAuthor, fizeram-se

necessários apenas quatro trechos de obra – por escritor – para alcançar valores muito próximos a 90% e até 100% de acerto em vários testes. Todos esses aspectos confirmam, portanto, a conformidade, contemporaneidade e viabilidade deste projeto na busca por novas alternativas para solucionar o problema em causa quando observados trabalhos afins já realizados no ambiente acadêmico.

3 MATERIAIS E MÉTODOS

Sendo este trabalho oriundo do *software* protótipo **DWE** (FILHO, 2010) – produzido pelo professor Me. Elenilson Vieira –, muitas tarefas iniciais foram dedicadas a um reconhecimento e estudo do projeto. Para direcionar o desenvolvimento e as melhorias implementadas no sistema, foram feitas pesquisas em fontes bibliográficas, essencialmente em artigos acadêmicos na área de linguística, a fim de entender a abrangência e nuances dos estudos linguísticos no campo da tecnologia da informação.

Primeiramente, a realização dessas pesquisas conferiu um conhecimento mais amplo sobre questões de ambas as áreas envolvidas, informática e linguística. Por conseguinte, foram revisados alguns conceitos básicos da gramática normativa da língua portuguesa, para o entendimento de quais aspectos gramaticais da língua seriam analisados pelo código.

Ademais, visando potencializar a realização de atividades e fazer o melhor uso do tempo, houve reuniões semanais com os orientadores, durante as quais se fez uma segmentação das atividades em tarefas menores que seriam cumpridas semanalmente. Por sua vez, foram utilizadas para a construção do sistema, obtenção ou armazenamento de dados e outras etapas, tecnologias posteriormente citadas, recomendadas pelos orientadores ou escolhidas após ponderação de seus prós e contras.

3.1 SCRUM

A metodologia Ágil Scrum foi utilizada desde o início do projeto, objetivando aprimorar a auto-organização dos participantes, além de melhorar a experiência de trabalho em equipe, motivação, bom relacionamento com prazos etc. (SABBAGH, 2014). Nessa perspectiva, o ciclo de trabalho adotado tem como objetivo uma boa entrega de resultados em um curto período de tempo. Neste projeto, ficou estabelecido um intervalo de uma semana (7 dias), sendo apresentado em cada reunião semanal (denominada *Sprint Planning*), o fluxo de trabalho feito, seus resultados, conclusões e encaminhamentos; por fim, eram definidas metas para a próxima semana. Tal formação metodológica de divisão de trabalhos semanais é definida como *Sprint*, conforme Sabbagh (SABBAGH, 2014): "O Sprint é o ciclo de desenvolvimento, onde o Incremento do Produto pronto é gerado pelo Time de Desenvolvimento a partir dos itens mais importantes do Product Backlog.". Sabbagh (SABBAGH, 2014) também explica o que é *Product Backlog*:

> É uma lista de tudo o que se acredita que será desenvolvido pelo Time de Desenvolvimento no decorrer do projeto. Em cada momento, essa lista é atualizada, ordenada de acordo com a importância para os clientes do projeto e possui apenas o nível de detalhes que é possível de se ter.

No desenvolvimento do WhoAuthor, as metas semanais eram divididas em duas vertentes: a primeira, no âmbito dos estudos linguísticos, em face dos quais estudamos a interação do projeto com a linguística e produzimos o presente relatório; já a segunda, por sua vez, é no âmbito da tecnologia da informação, a partir da qual programamos o sistema e arquitetamos o *corpus*. Portanto, o *SCRUM* foi de importância imprescindível para o desenvolvimento deste trabalho.

3.2 GIT E GITHUB

É indeclinável que todos os dados programáveis do sistema sejam armazenados de forma segura para que não haja perdas consideráveis para o projeto. Pensando nessa problemática, é comumente utilizado um Sistema de Versionamento de Código, onde várias versões do mesmo código são salvas e seu histórico é armazenado em um servidor remoto, garantindo, assim, segurança e transparência.

No desenvolvimento do WhoAuthor tanto no back-end quanto no protótipo do front-end foi utilizado o Git, um Sistema de Versionamento de Código *Open Source* e gratuito. Criado por Linus Torvalds, o Git garante imutabilidade, transações atômicas, integridade e confiança (LOELIGER; MCCULLOUGH, 2012), e foi justamente por essas garantias que ele foi escolhido para ser usado neste trabalho.

Para um funcionamento conjunto com o Git, o GitHub (GITHUB, 2021) foi utilizado para hospedar os códigos fonte dos projetos (*back-end* e *front-end*)[1]. A utilização do GitHub facilita a interação entre os desenvolvedores deste trabalho, possibilitando uma colaboração conjunta entre eles.

3.3 JAVA

Java é uma linguagem de programação orientada a objetos, que foi criada na década de 90, desde quando vem sendo modernizada, sendo ainda hoje considerada uma das mais usadas linguagens de programação do mundo (JAVA, 2021). Dentre diversas características dessa poderosa linguagem de programação, algumas se sobressaem e fazem jus aos motivos pelos quais Java foi escolhida para ser a principal linguagem de programação deste trabalho, usada no back-end em sua totalidade.

Graças ao *Java Virtual Machine* (JVM), o Java tem capacidade de ser executado em qualquer aparelho eletrônico. Uma vez compilado, a JVM gera um arquivo *bytecode* específico para o sistema operacional no qual programa está hospedado; em resumo, um único código é compilado para que seja executado em diversas plataformas diferentes.

[1] Repositório do back-end disponível em: <https://github.com/pablodeyvid11/Attribution_of_authorship-Backend> e do front-end disponível em: <https://github.com/pablodeyvid11/Attribution_of_authorship-FrontWeb>

Ademais, o Java possui o *Garbage Collector*, que nada mais é do que seu sistema de gerenciamento automático de memória gerenciado pela JVM. Dessa maneira, o *Garbage Collector* remove automaticamente os objetos alocados em memória que não estão sendo mais utilizados pela aplicação, fazendo com que menos memória seja utilizada durante a execução. Esse sistema é essencial para o WhoAuthor, pois como uma grande quantidade de dados e objetos são utilizados para fazer um treinamento ou atribuição, é de extrema importância que os dados que se tornaram obsoletos sejam descartados para que haja um melhor desempenho e aproveitamento do sistema.

Por fim, um dos maiores motivos pelo qual Java foi escolhido para ser o código base do back-end do WhoAuthor é sua fácil implementação de *Threads*. Uma *Thread* é um "fio de execução", ou seja, é um processo rodando no computador de uma máquina (OAKS; WONG, 1999). O *multithreading* é a capacidade de um sistema de executar diferentes *threads* simultaneamente; o uso dessa abordagem no nosso trabalho permite, assim, que diferentes textos sejam analisados ao mesmo tempo, diminuindo seu tempo de processamento; também é utilizado para que não ocorram problemas como *Time Limit Request Exceeded*, quando o tempo limite de uma requisição para o back-end é excedido.

3.4 WEB SERVICE RESTFUL COM SPRING BOOT

O back-end do WhoAuthor é um servidor *RESTful*, ou seja, um *web service* simples, implementando métodos HTTP e os princípios de REST, que segundo Roy Fielding (FIELDING, 2000):

> A REST é pretendida como uma imagem do design da aplicação se comportará: uma rede de websites (um estado virtual), onde o usuário progride com uma aplicação selecionando as ligações (transições do estado), tendo como resultado a página seguinte (que representa o estado seguinte da aplicação) que está sendo transferida ao usuário e apresentada para seu uso.

O protótipo do front-end do WhoAuthor faz uma requisição ao back-end, requisição essa que é processada, sendo devolvida uma resposta adequada e imutável à própria requisição. Três aspectos são desenvolvidos em um servidor RESTful, a saber: primeiro, uma URI raiz para o serviço e seus métodos (por exemplo: "/training/advanced"); depois, tipos MIME suportados para as requisições (o utilizado foi o *JavaScript Object Notation JSON* (JSON, 2021)); por último, a implementação de métodos HTTP: PUT, GET, POST e DELETE (FERREIRA; JR, 2018).

Considerando os requisitos para que uma aplicação seja RESTful apresentados anteriormente, a dificuldade em implementar uma solução eficiente e não prolixa é crescente. Pensando nisso, foi utilizado o ecossistema Spring com o apoio das ferramentas do Spring

Boot, que, de maneira simples, rápida e de fácil entendimento, entrega toda a segurança do framework junto com todas as funcionalidades já pré-implementadas (SPRING, 2021).

O Spring Boot utiliza Maven, que é uma estrutura de gerenciamento de projeto baseada em padrões e código aberto que simplifica a construção, teste, relatório e empacotamento de projetos (VARANASI, 2019).

3.5 SPRING DATA JPA

O Spring Data JPA é um conjunto de bibliotecas pertencentes ao Framework Spring, cujo objetivo é a fácil implementação de métodos de acesso a banco de dados (JPA, 2021). Utilizando camadas de acesso a dados baseado no Java Persistence API, JPA, o Spring Data JPA permite um acesso ao banco de dados de forma simples, limpa e dinâmica.

Utilizando o padrão de projeto *Model-view-controller* (MVC) (DING; LIU; TANG, 2012), é implementada apenas uma interface que estende a classe JpaRepository, que usa como parâmetro a entidade tabela do banco e o tipo de ID utilizado. Esse padrão faz com que o sistema possua menos classes, consequentemente com melhor entendimento e de fácil programação.

3.6 POSTGRESQL

O sistema gerenciador de banco de dados (SGBD) escolhido foi o PostgreSQL (POSTGRESQL, 2021), que utiliza o método objeto-relacional e armazena, por meio de tabelas, as instâncias dos autores cadastrados e das palavras pesquisadas. Gratuito e *open source*, o PostgreSQL é um dos SGBD mais utilizados no mundo, por causa da sua robustez, do seu poder e da sua excelente comunidade. Segundo Milani (MILANI, 2008):

> O PostgreSQL encontra-se em uma versão perfeitamente estável e confiável, com os principais recursos existentes nos bancos de dados pagos disponíveis no mercado. Suas capacidades são para suprir as necessidades de pequenas, médias e grandes aplicações.

Uma das vantagens de usar o PostgreSQL neste trabalho é que ele não possui limite de tamanho em seus bancos de dados, e o limite para as tabelas é extremamente alto na escala de 32 terabytes cada. Um limite alto é de suma importância, pois a tabela de palavras do WhoAuthor pode conter na base de 400 mil linhas e pode chegar a 1 milhão ou mais.

3.7 SPRING TOOLS SUITE 4 E VISUAL STUDIO CODE

O Spring Tools Suite 4, STS, foi o Ambiente de Desenvolvimento Integrado (IDE, do inglês *Integrated Development Environment*) utilizado para o desenvolvimento do back-end

do WhoAuthor. Por ser um IDE com diversas ferramentas e altamente integrada com o Spring Boot, ele foi a melhor ferramenta possível para que o sistema fosse desenvolvido de forma prática e completa (SPRINGTOOLS, 2021). O STS é baseado no Eclipse (ECLIPSE, 2021), uma das maiores IDEs utilizadas para a programação em JAVA.

Já para o desenvolvimento do protótipo do front-end do WhoAuthor, foi utilizado o editor de código-fonte Visual Studio Code (VSCODE, 2021), popularmente conhecido como VS code, desenvolvido pela Microsoft. O VS code é gratuito, open source e graças a sua biblioteca de extensões bastante abrangente, nele é possível programar em qualquer linguagem, além de possuir IntelliSense, que é definida pela Microsoft (MICROSOFT, 2019a) como:

> O IntelliSense é uma ajuda de preenchimento de código que inclui inúmeras funcionalidades: Listar Membros, Informações do Parâmetro, Informações Rápidas e Completar Palavra. Essas funcionalidades ajudam você a aprender mais sobre o código que está usando, a manter o acompanhamento dos parâmetros que está digitando e a adicionar chamadas a métodos e propriedades pressionando apenas algumas teclas.

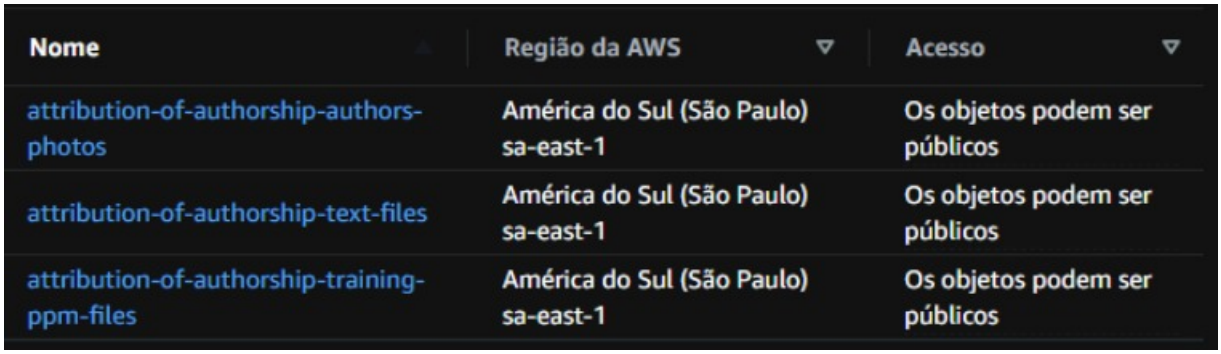
3.8 AMAZON AWS E AMAZON SIMPLE STORAGE SERVICE (S3)

Uma gama de arquivos são necessários para que o WhoAuthor funcione, dentre os quais estão os arquivos ".txt"que o usuário disponibiliza para realizar o treinamento do sistema, as fotos usadas para representar cada autor e os arquivos .training-ppm que são gerados quando um autor é treinado. Pensando nisso, foi levantada a problemática de como esses arquivos seriam armazenados de forma segura e que não acarretaria problemas futuros. Pensando nisso, o *Amazon Simple Storage Service*, o S3, do Amazon AWS (AWS, 2021) foi escolhido. O S3 é um serviço de armazenamento de arquivos de forma remota, que permite armazenar ilimitados arquivos em seus servidores. Tendo um sistema robusto, seguro e de fácil acesso, o Amazon S3 é uma das melhores escolhas do mercado na área de storage.

A arquitetura do S3 é simples, tendo apenas dois componentes principais: os objetos e os *buckets*. Os objetos são os próprios arquivos, que serão guardados nos servidores da Amazon; já os *buckets* são *containers* que são utilizados para armazenar os objetos. Essa arquitetura simples significa que o S3 é um sistema flexível, fácil e que se adapta às diferentes finalidades do mercado (MURTY, 2008).

No WhoAuthor a arquitetura de arquivos foi dividida em três *buckets*, que são responsáveis por armazenar as fotos dos autores, seus arquivos utilizados para o treinamento e os modelos matemáticos dos autores, os .training-ppm, como é representado na figura 1. Uma vez feito o upload no S3, é gerado um link de acesso, que é armazenado em sua respectiva coluna na instância do autor no banco de dados.

Figura 1 – Buckets do WhoAuthor no Amazon S3



Nome	Região da AWS	Acesso
attribution-of-authorship-authors-photos	América do Sul (São Paulo) sa-east-1	Os objetos podem ser públicos
attribution-of-authorship-text-files	América do Sul (São Paulo) sa-east-1	Os objetos podem ser públicos
attribution-of-authorship-training-ppm-files	América do Sul (São Paulo) sa-east-1	Os objetos podem ser públicos

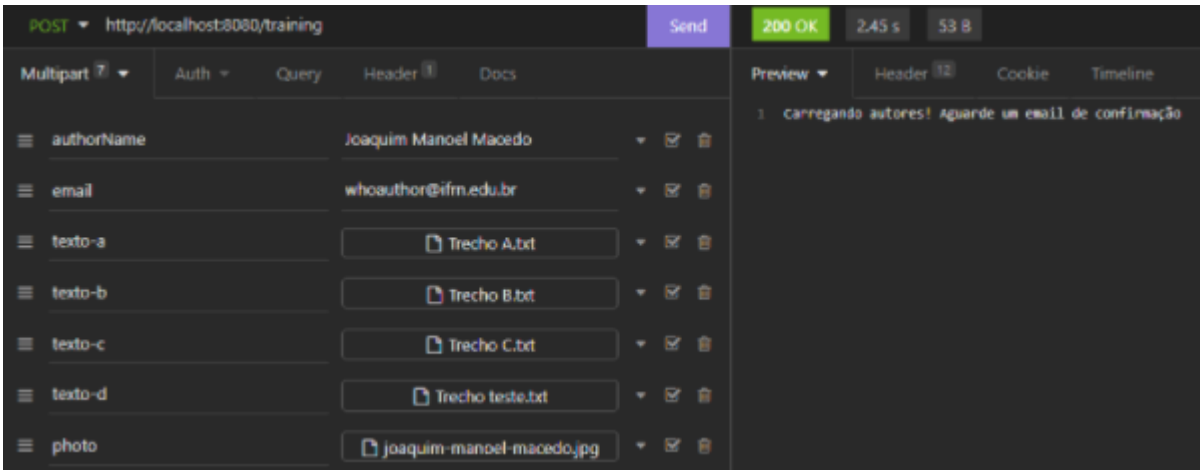
Fonte: Autoria própria

3.9 INSOMNIA REST

O Insomnia é um poderoso cliente *API REST* (INSOMNIA, 2021), que é capaz de enviar requisições em *multipart form*, *GraphQL Query*, *JSON*, *XML*etc. Além disso, ele gerencia de forma eficiente *cookies* e variáveis de ambiente e é um *open source* gratuito disponível para Linux, MAC e Windows (STACKSHARE, 2016).

O uso do Insomnia no desenvolvimento do WhoAuthor fez-se necessário, pois com ele eram feitos os testes no servidor RESTful desenvolvido no back-end. A figura 2 mostra um exemplo de requisição POST para treinamento de um autor no sistema.

Figura 2 – Requisição para treinamento



Fonte: Autoria própria

3.10 FIGMA

A fim de criar um modelo para o protótipo do front-end do WhoAuthor que respeitasse padrões de UI (Interface de usuário) e UX (Experiência de usuário), o Figma foi escolhido. Gratuito e com uma imensa comunidade, o Figma é um editor gráfico de

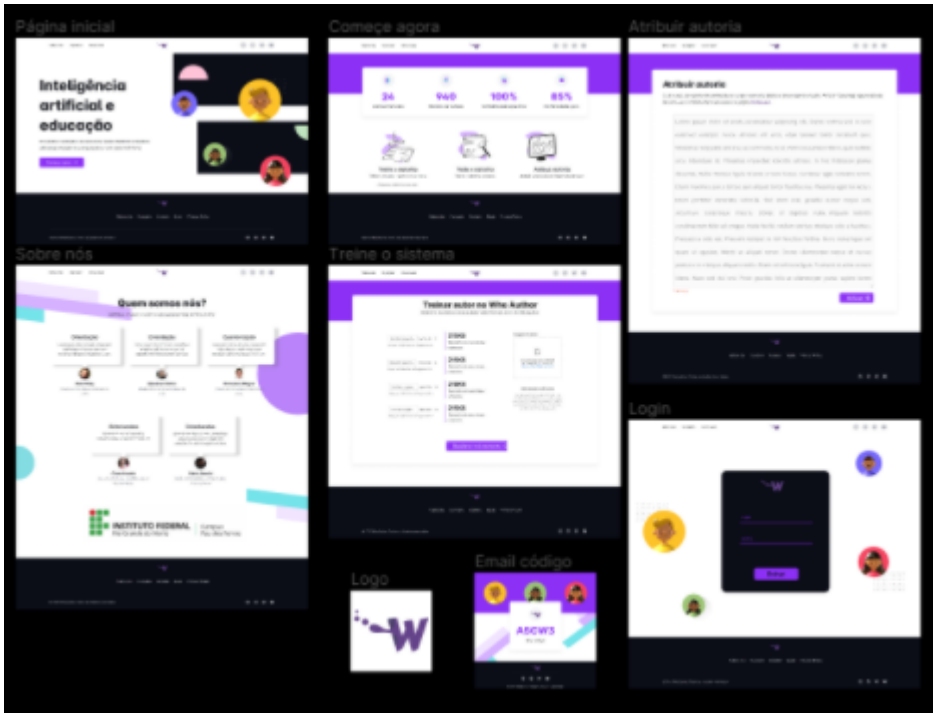
prototipagem de projetos, que tem como objetivo criar designs e modelos, comumente usado no desenvolvimento de sites (FIGMA, 2021).

Todas as telas do protótipo do front-end e a logo do WhoAuthor foram criadas no Figma, o que significa que ele é o principal editor utilizado em todo o processo de desenvolvimento. A figura 3 mostra a logo do WhoAuthor e a figura 4 mostra algumas das inúmeras telas produzidas no Figma[2].

Figura 3 – Logo do WhoAuthor

Fonte: Autoria própria

Figura 4 – Telas produzidas no Figma



Fonte: Autoria própria

[2] Projeto disponível em: <https://www.figma.com/file/xzA1dBv4r3WYOdYxjULYZj/TCC?node-id=0%3A1>

3.11 TYPESCRIPT

O TypeScript é uma linguagem de programação *open source* que se baseia no JavaScript, uma das tecnologias mais usadas no mundo (JAVASCRIPT, 2021). A principal diferença entre o TypeScript e o JavaScript é que eles possuem tipagens completamente diferentes. Enquanto o JavaScript é de tipagem dinamicamente fraca, ou seja, o conceito de "tipo"é superficial, o TypeScript possui uma tipagem estática; logo os "tipos"dos objetos terão que ser seguidos à risca, se assemelhando com o Java.

O TypeScript possui a mesma sintaxe e semântica que os desenvolvedores JavaScript conhecem hoje; porém, devido a sua tipagem, é necessário definir o tipo de cada objeto na sua declaração, o que é feito por meio de blocos de *interface* ou *type*. Uma vez escrito, o código TypeScript passa por um processo chamado de *transpilação*, que consiste na sua transformação em código JavaScript, para assim ser executado pelo *browser* (MICROSOFT, 2019b).

A principal vantagem do TypeScript é a possibilidade de descobrir erros durante o desenvolvimento e incrementar a inteligência (IntelliSense) do Visual Studio Code. Além de ajudar no ambiente de desenvolvimento, o TypeScript ainda permite um melhor entendimento do código e possibilita ao desenvolvedor uma melhor integração com a plataforma (ROCKETSEAT, 2021).

3.12 REACT.TS

O React.ts é uma biblioteca para a criação de interfaces de usuário em TypeScript (REACTTS, 2021). Por ter uma grande comunidade e ter um alto suporte às necessidades básicas, o React.ts foi escolhido para ser usado na produção do protótipo do front-end do WhoAuthor.

A interface das aplicações produzidas em React.ts é baseada na ideia de componentes, que são trechos de códigos que representam blocos na interface desenvolvida: *NavBar*, *Footer*, *Cards* etc. O uso de componentes permite a reutilização de código, diminuindo sua sintaxe e aumentando seu entendimento.

4 RESULTADOS

4.1 PROCESSO DE ESTRUTURAÇÃO DO *CORPUS*

Inicialmente, vale destacar o conceito de *corpus* no âmbito deste trabalho. De origem latina, *corpus* significa "corpo", e, segundo Bauer e Aarts (BAUER; AARTS, 2010), caracteriza-se como todos os materiais que fundamentam, no caráter científico, um trabalho ou tese. Na criação do WhoAuthor, o corpus é composto por todos os textos dos autores que foram utilizados para o treinamento e testagem do sistema, objetivando estabelecer, de maneira confiável e com base científica e firmada na realidade, taxas de assertividades consistentes – a lista de autores aqui utilizados foi inspirada na seleção de autores que Barufaldi (BARUFALDI et al., 2010) aplicou em seu estudo.

4.2 PROCESSO DE TREINAMENTO DO SISTEMA

O sistema realiza o processo de treinamento em dois parâmetros diferentes: o primeiro leva em consideração o tamanho do contexto, que varia de 2 a 10; e o segundo tem como base o tamanho do arquivo resultante de 48KB e 200KB.

Para iniciar o processo de treinamento, o usuário disponibiliza quatro arquivos de texto que possuem fragmentos de livros do autor a ser treinado (dispostos no *corpus*). Esses textos são nomeados em: Texto A, Texto B, Texto C e Texto D, a fim de utilizar o método de validação cruzada (EBRAHIMPOUR et al., 2013) para unificá-los e criar o arquivo resultante. Tal método possibilita que mesmo com apenas 96 textos, o corpus tenha, na prática, quatro vezes mais material a ser analisado, devido às baterias de treinamento e teste, que se alternam, possibilitando, assim, que a amostra de texto disponível possa ser explorada ao máximo. O Apêndice A, mostra a relação entre os livros treinados, os livros de teste e os dois parâmetros apresentados anteriormente de um autor X — processo esse que é repetido para todos os demais autores.

A unificação (pré-processamento) dos textos, por sua vez, é feita de forma automática pelo sistema, pois ele faz um recorte apenas dos dados necessários, sejam eles com 16KB, a fim de produzir o arquivo de 48KB, ou com 67KB, para produzir o arquivo de 200KB.

4.3 PROCESSO DE ATRIBUIÇÃO DE AUTORIA

A partir do *corpus* pré-processado é gerado um novo arquivo com a sequência de classificações gramaticais do texto, que será usado no treinamento do sistema. Através da API do Java HttpClient, as consultas, para as classificações gramaticais de cada palavra

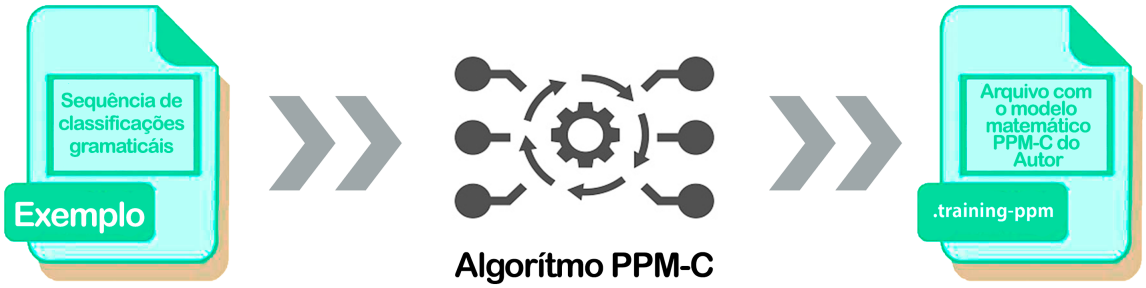
do texto, são feitas no dicionário online Priberam (PRIBERAM, 2021). Assim, é finalizado o processo de codificação do texto.

Em seguida, esse arquivo passa por um processo de compactação e, à medida que é compactado, é construída a representação do modelo estatístico do autor, com a ajuda de um programa criado em Java. Ao final desse processo, é obtido um arquivo training-ppm, que contém o modelo matemático correspondente às características individuais do autor. Depois que o arquivo training-ppm é gerado, é criada uma instância do autor em questão, na qual são referenciados o arquivo e as variações apresentadas na tabela 1, e depois adicionado no banco de dados. A figura 5 é a representação do que foi explicado neste parágrafo.

Tabela 1 – Estrutura da validação cruzada

Bateria	Livro usados no treinamento	Livros usados no teste
1ª	A, B e C	D
2ª	A, B e D	C
3ª	A, C e D	B
4ª	B, C e D	A

Figura 5 – Processo de formação do modelo matemático referente a cada autor

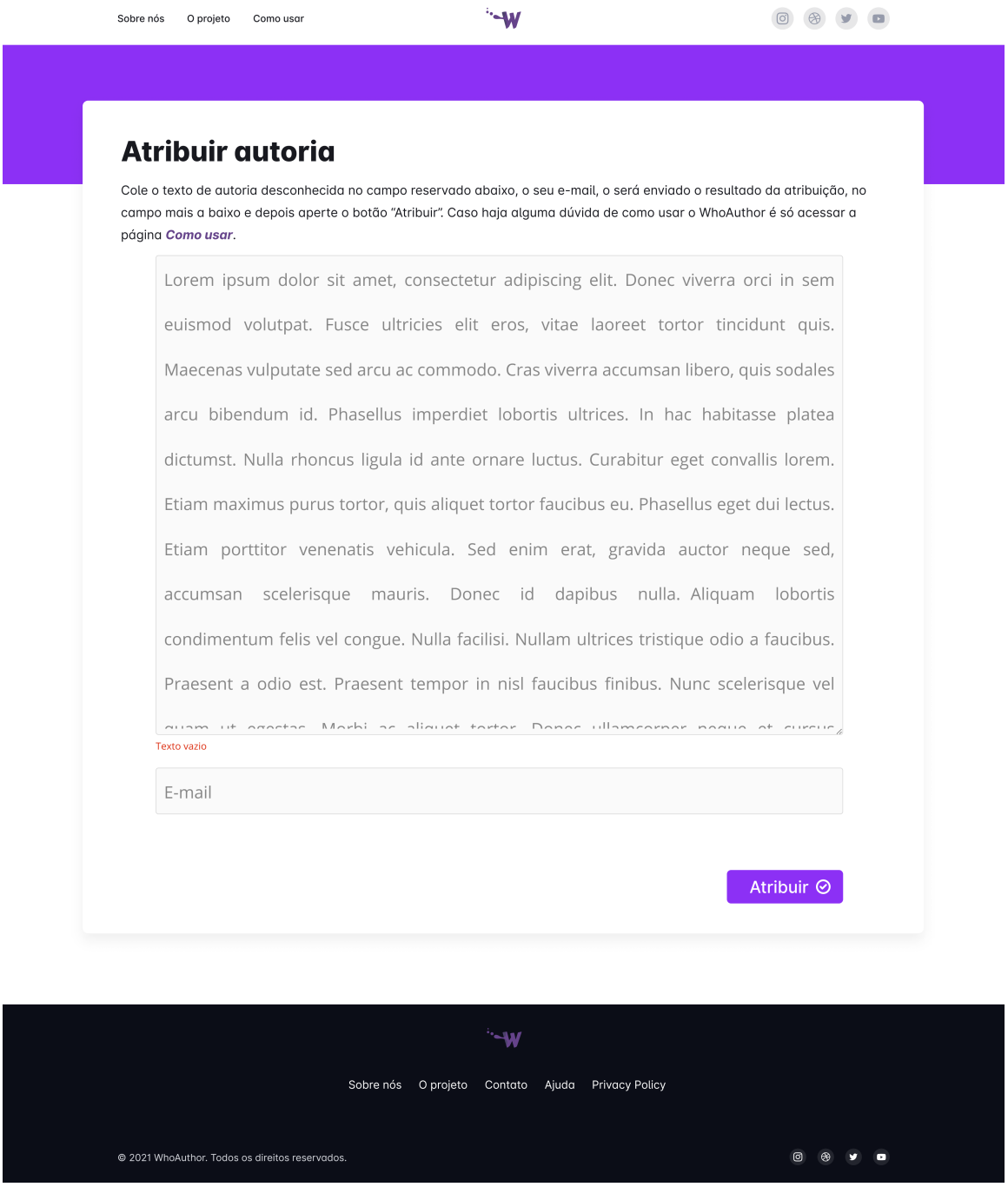


Fonte: Autoria própria

Para atribuir autoria, o usuário tem que disponibilizar um fragmento de texto de autoria desconhecida e seu e-mail para o sistema: de maneira muito simples, ele apenas coloca os dados pedidos em *HTML inputs* e inicia o processo de atribuição, clicando no botão “Atribuir”, como é mostrado na figura 6.

Esse fragmento é convertido em sequências de classificações gramaticais e comprimido com cada um dos modelos de autores que o sistema tem em seu banco de dados; o tamanho de cada arquivo resultante é, em seguida, armazenado e o que for menor é apontado pelo sistema como o mais provável de ser o autor do texto desconhecido (a

Figura 6 – Tela de atribuição de autoria no *front-end* do WhoAuthor



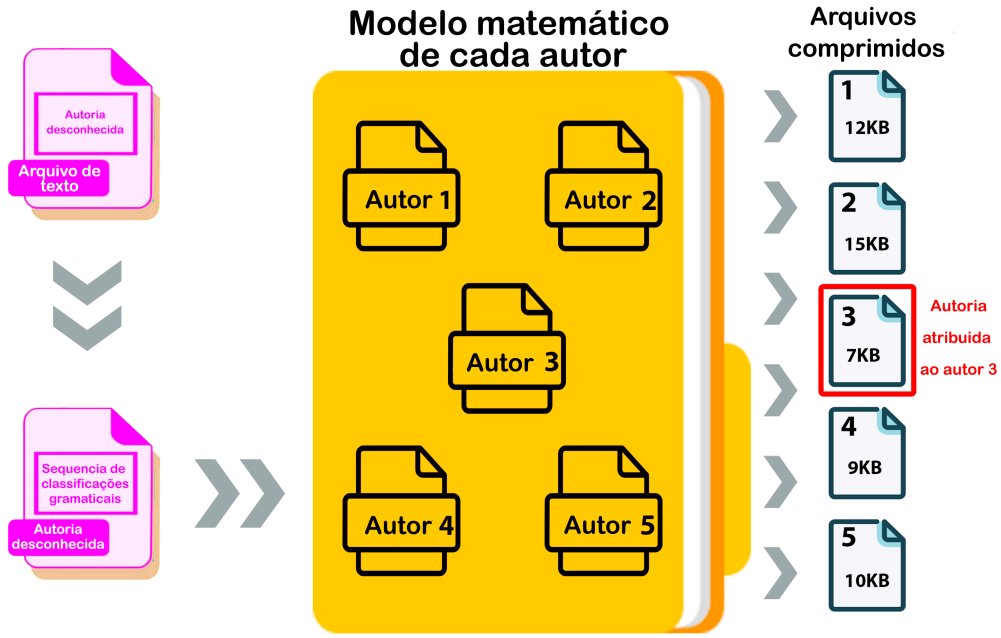
Fonte: Autoria Própria

figura 7 ilustra esse processo). Finalizada a atribuição de autoria, o sistema encaminha um e-mail para o usuário, notificando-o do resultado da atribuição.

4.4 MODIFICAÇÕES NO CÓDIGO

O WhoAuthor provém do software DWE, que foi desenvolvido pelo professor Me. Elenilson Vieira em uma das disciplinas de seu mestrado (FILHO, 2010). O DWE era um

Figura 7 – Processo de atribuição de autoria



Fonte: Autoria Própria

software, produzido em Java e com uma interface embutida — também programada em Java, utilizando as bibliotecas do Java Swing.

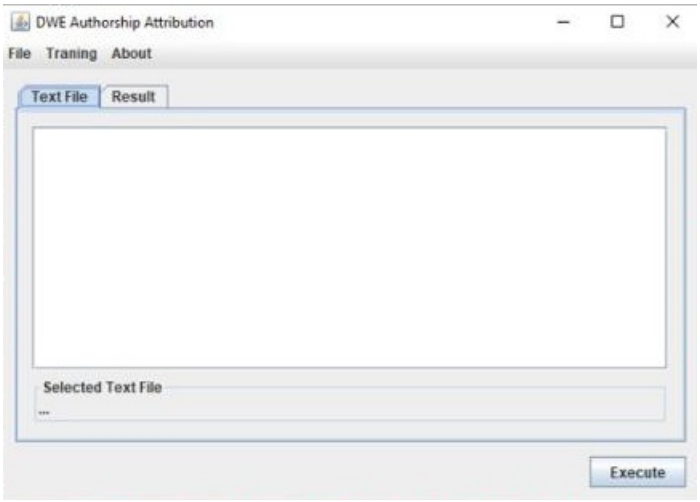
4.4.1 PROCESSO INICIAL

Inicialmente, o funcionamento do DWE estava totalmente prejudicado por problemas como bibliotecas *deprecated*[1], que era o caso das que faziam requisições no dicionário online Priberam. Também foi notado que a forma de acesso ao banco de dados estava igualmente obsoleta e que todas as bibliotecas que faziam esse trabalho foram descontinuadas. Por isso, foi necessária uma análise de todas as classes e arquivos de seu código fonte e, a posteriori, a realizalção das devidas adaptações.

No afã de resolver os problemas iniciais no código, fazer com que ele funcione de maneira correta e tentar evitar problemas futuros, foi utilizada a ferramenta Maven (MAVEN, 2021). O uso do Maven permitiu que a própria aplicação fosse responsável por realizar o download das bibliotecas utilizadas, de forma dinâmica e sempre buscando as mais atuais, ou seja, evitando problemas futuros de obsolescência. Com as bibliotecas atualizadas e com o código adaptado para a utilização do Maven, o DWE voltou a funcionar (as figuras 8 e 9 a seguir mostram como era a interface da aplicação).

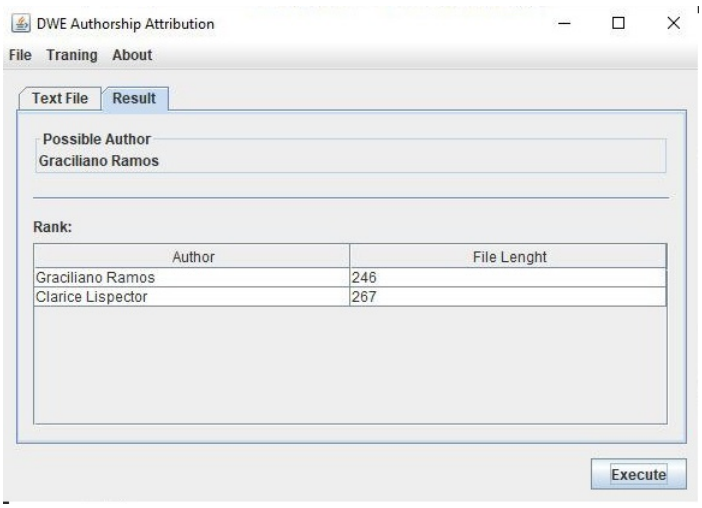
[1] Segundo o dicionário de língua inglesa da Universidade de Cambridge (PRESS, 2021), *deprecated* é algo que não deve ser mais usado. É um recurso ainda existente, mas que é considerado defasado e tem algo melhor para usar. Pode ser traduzido para "Obsoleto"ou "Descontinuado", mas nenhuma dessas palavras expressa bem o seu significado.

Figura 8 – Tela inicial do DWE



Fonte: Autoria Própria

Figura 9 – Tela de resultados do DWE



Fonte: Autoria Própria

4.4.2 OTIMIZAÇÃO DO DWE

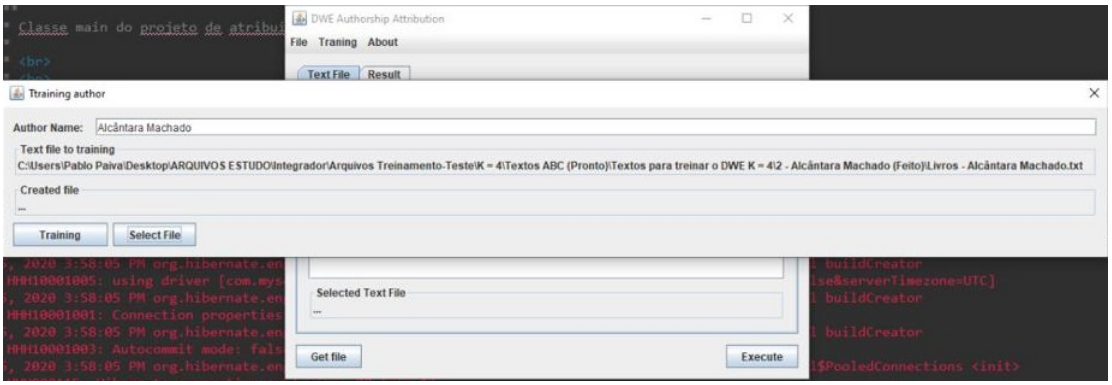
Algumas das funcionalidades do DWE eram completamente manuais. Com o fito de automatizar alguns métodos, o software passou por algumas modificações.

O processo de treinamento de um autor consiste em disponibilizar um arquivo .txt com todos os fragmentos de livros já unificados, juntamente com o nome do autor a ser treinado. Esse procedimento tinha que ser repetido levando em consideração todos os parâmetros já aqui apresentados, a saber: o autor, o contexto, tamanho do arquivo unificado e formato de unificação — Textos “ABC”, “ABD”, “ACD” e "BCD". Sendo o processo bastante lento, o primeiro autor demorou 27 horas para ser treinado; em seguida, como o banco de dados já estava mais populado com novas palavras, esse tempo diminuiu,

chegando a 30 minutos.

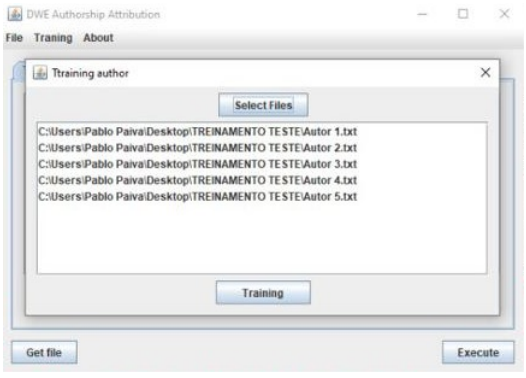
Mesmo com o tempo de treinamento reduzido, ainda se torna bastante cansativo para o usuário treinar todos os autores em sequência, exigindo que ele passe dias em frente ao computador, devido à grande quantidade de autores utilizados neste trabalho. Com o objetivo de tentar melhorar a experiência do usuário e a forma como era feito o treinamento, foi realizada a adaptação que permitia que fossem treinados vários autores ao mesmo tempo. Ele tinha, para isso, apenas que disponibilizar todos os arquivos unificados, que o sistema do DWE iria treinar os respectivos autores — o nome do autor, era o nome do arquivo (a figura 10 mostra como era o processo antes de ser feita as modificações; já as figuras 11 e 12 ilustram o mesmo processo após as modificações).

Figura 10 – Tela de treinamento de autor antiga



Fonte: Autoria Própria

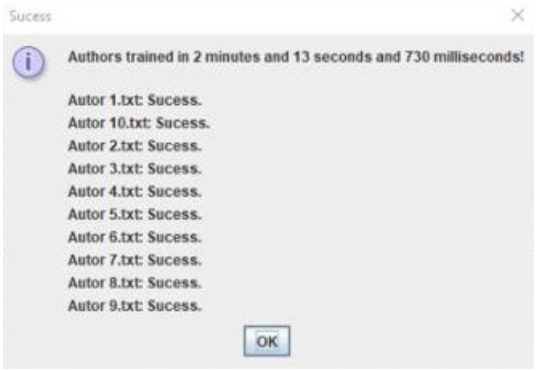
Figura 11 – Tela de treinamento de autor melhorada



Fonte: Autoria Própria

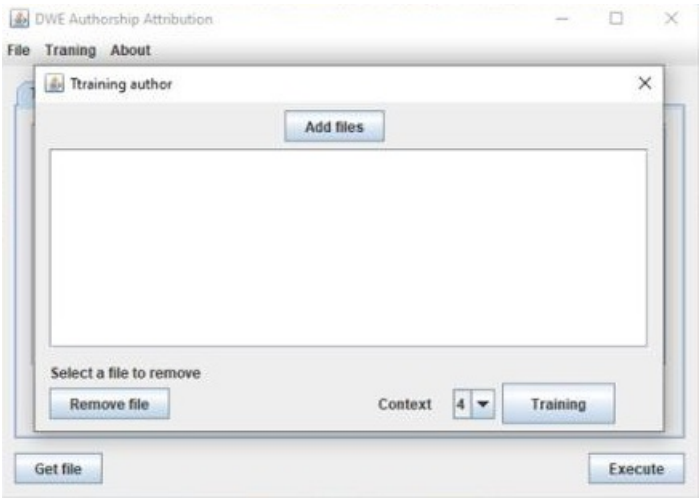
Um outro ponto observado é que, para alterar o contexto de treinamento, era necessário alterar manualmente o código dentro da classe AbstractPPMModule. Uma solução para esse problema foi a inserção de um botão que permitia a escolha do contexto pelo usuário (a figura 13 mostra como ficou o processo de treinamento após essa inserção, além de um botão adicional, que remove algum arquivo selecionado por engano).

Figura 12 – Confirmação de treinamento



Fonte: Autoria Própria

Figura 13 – Tela de trinamento após a inclusão do combobox que seleciona o contexto



Fonte: Autoria Própria

No entanto, um problema pontual do DWE é que, quando ele faz o processo de codificação dos textos, transformando-os em sequências de classificações gramaticais, a pontuação é descartada e apenas as palavras literais são consideradas.Assim, objetivando adicionar as pontuações nas sequências de classificações gramaticais, foi feita a inclusão delas no tipo enumerado (enum) responsável por nomear os elementos textuais — palavras e, agora, pontuações — presentes nos textos (a figura 14 mostra a implementação desse enum, que atualmente nomeia 29 elementos textuais diferentes).

4.4.3 CRIAÇÃO DO WHOAUTHOR

O DWE, por mais que tenha sofrido diversas alterações e atualizações, ainda apresentava alguns problemas, como a abordagem antiga do Java, em consequência da qual é acarretada a presença de código prolixo e com algumas repetições. Com o objetivo de usar uma abordagem mais moderna do Java e, consequentemente, ter um código

Figura 14 – Enum WordClassification

```java
package br.edu.ifrn.entities.enums;

import java.io.Serializable;

public enum WordClassification implements Serializable {
    NOUN(0, "Substantivo"),
    ARTICLE(1, "Artigo"),
    PRONOUN(2, "Pronome"),
    VERB(3, "Verbo"),
    ADJECTIVE(4, "Adjetivo"),
    CONJUNCTION(5, "Conjunção"),
    INTERJECTION(6, "Interjeição"),
    PREPOSITION(7, "Preposição"),
    ADVERB(8, "Advérbio"),
    FINAL_PONCTUATION(9, "Ponto final"),
    VIRGULA(10, "Vírgula"),
    EXCLAMACAO(11, "Ponto de exclamação"),
    INTERROGACAO(12, "Ponto de interrogação"),
    RETICENCIAS(13, "Reticencias"),
    ASTERISCO(14, "Asterisco"),
    ABRE_PARENTESES(15, "Abre Parenteses"),
    FECHA_PARENTESES(16, "Fecha Parenteses"),
    ABRE_CHAVES(17, "Abre Chaves"),
    FECHA_CHAVES(18, "Fecha Chaves"),
    ABRE_COLCHETES(19, "Abre Colchetes"),
    FECHA_COLCHETES(20, "Fecha Colchetes"),
    A_ORDINAL(21, "A - Ordinal"),
    O_ORDINAL(22, "O - Ordinal"),
    PONTO_VIRGULA(23, "Ponto e vírgula"),
    DOIS_PONTOS(24, "Dois Pontos"),
    APOSTROFE(25, "Apóstofre"),
    ASPAS(26, "Aspas"),
    INTERMEDIATE_PONTUATION(27, "Pontuação intermediária"),
    UNKNOWN(28, "Desconhecido");

    private WordClassification(int id, String classificacao) {
        this.id = id;
        this.classificacao = classificacao;
    }

    public Integer getId() {
        return this.id;
    }

    public String getClassificacao() {
        return this.classificacao;
    }

    private Integer id;
    private String classificacao;
}
```

Fonte: Autoria Própria

mais limpo e com mais funcionalidades, o WhoAuthor foi criado — inicialmente com o desenvolvimento do *back-end*.

4.4.4 PRIMEIRO ESTÁGIO DE DESENVOLVIMENTO: ADAPTAÇÃO

Utilizando o STS, foi criado um Spring Started Project, um projeto base de um web service RESTful em Java, tudo isso dentro do ecossistema Spring. Inicialmente, todas as classes do DWE foram copiadas e coladas dentro do projeto recém-criado; foram feitas, então, as devidas importações e o sistema do DWE já estava funcionando normalmente, porém dentro do SSP.

No contexto desses procedimentos, fez-se necessária a transformação do projeto Java Swing em Web Service RESTful. Em seguida, foi utilizado o padrão de projeto

Model View Controller (MVC), que consiste em subdividir o projeto em três camadas: a Model, que é responsável pela manipulação de dados (os denominados repositories, que são interfaces que permitem o acesso ao banco de dados); a Controller, responsável pelas regras de negócio do sistema (as classes services e as demais classes que exercem as funções do programa); e, por fim, a View, que consiste em classes que interagem com o usuário — no back-end do WhoAuthor, a View foi implementada por meio de uma Application Program Interface (API) na forma de um Web Service RESTful)(DEACON, 2009).

As primeiras classes a serem implementadas foram as repositories: AuthorRepository e WordRepository, que são responsáveis por fazer a interação com o banco de dados, nas tabelas dos autores e das palavras. Sua implementação é bastante simples, pois as interfaces estendem o JpaRepository — uma classe do ecossistema Spring —, fazendo com que todos os métodos básicos já venham pré-implementados.

Uma funcionalidade bastante útil presente no JpaRepository é a possibilidade de implementar *Query Methods* que, apenas por sua definição de nome, têm sua funcionalidade implementada pelo ecossistema Spring. Por exemplo: um método declarado "findByName" tem seu código já implementado pelo Spring, buscando objetos no banco de dados levando em consideração o parâmetro "name".

Também é possível declarar métodos a partir da escrita de uma Query em JPQL, do inglês Java Persistence Query Language (a figura 15 é um exemplo de declaração de um repository).

As classes referente à camada service, inicialmente, são apenas quatro: AuthorService e WordService, que são responsáveis por implementar as regras de negócio para, respectivamente, os autores e as palavras; o AttributionOfAuthorshipService, responsável por fazer a atribuição de autoria dos textos disponibilizados; e por fim, o TrainingService, responsável por realizar o treinamento do sistema.

Por último, existe a implementação dos controllers, que são classes que possuem endpoints, ou seja, métodos finais que, a partir de uma requisição HTTP dada em uma URI raiz, retornam uma resposta imutável aos parâmetros da requisição.

Muitos endpoints foram criados, a exemplo do "/training", que, a partir de uma requisição POST e parâmetros multiform, faz o treinamento de um autor (a figura 16 mostra esse método).

4.4.5 SEGUNDO ESTÁGIO DE DESENVOLVIMENTO: AUTOMAÇÃO

A organização dos fragmentos de textos que são usados no treinamento dos autores era completamente manual, um trabalho bastante cansativo e repetitivo. Para cada autor, foram baixados quatro livros diferentes, três para usar no treinamento e um para ser usado no teste desse autor. De forma manual, foram selecionados fragmentos de texto desses

Figura 15 – Repository Class AuthorRepository

```java
package br.edu.ifrn.repositories;

import java.util.List;
import java.util.Optional;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.stereotype.Repository;

import br.edu.ifrn.entities.Author;

@Repository
public interface AuthorRepository extends JpaRepository<Author, Long> {

    Optional<Author> findById(Long id);
    List<Author> findByAuthorName(String authorName);
    List<Author> findByFormat(String format);
    List<Author> findByK(Integer K);
    List<Author> findBySizeTextTrainFile(Integer sizeTextTrainFile);

    @Query("select u from Author u where u.sizeTextTrainFile = ?1 and u.format = ?2 and u.k = ?3")
    List<Author> findByFields(Integer sizeTextTrainFile, String format, int k);
}
```

Fonte: Autoria Própria

Figura 16 – EndPoint training

```java
@PostMapping(value = "/training")
public ResponseEntity<AuthorDTO> treinar(@RequestBody TrainingBody tb) {

    AuthorDTO author =
        service.train(
            tb.getAuthorName(), tb.getK(),
            tb.getTexto(), tb.getFileSize(),
            new WordClassifier(wordRepo), tb.getFormat());

    URI uri = service.getURI(TypeUploadS3.TRAINING_PPM_FILE, author.getAuthorName() + ".training-ppm");

    return ResponseEntity.created(uri).body(author);
}
```

Fonte: Autoria Própria

livros e colocados em arquivos txt, que tinham 16KB ou 67KB. Esses fragmentos foram unificados de acordo com a tabela 1, mostrada na sessão 4.3.

O trabalho manual para realizar o treinamento de um autor demandava, assim, bastante tempo dos desenvolvedores, razão pela qual houve a necessidade de automatizar

esse processo, otimizando o processo de treinamento de forma geral.

O endpoint "/training" recebia apenas as informações de um treinamento na forma de JSON, contendo o contexto, formato, tamanho e autor específicos. Depois da implementação da automação, o mesmo endpoint passou a receber os quatro arquivos de texto — Texto "A", "B", "C" e "D" e o nome do autor. Com essas informações, é realizado o treinamento do autor em todas as possibilidades possíveis, de acordo com o Apêndice A

4.4.6 TERCEIRO ESTÁGIO DE DESENVOLVIMENTO: NOVAS FUNCIONALIDADES

Duas novas funcionalidades importantes foram adicionadas para integrar modernidade e seriedade ao WhoAuthor: a primeira delas é a implementação de um serviço de e-mail para comunicação com o cliente; a segunda é a utilização do Amazon Simple Storage Service, Amazon S3, para o armazenamento de arquivos.

A API funciona por meio de requisições HTTP, e essas requisições necessitam de respostas rápidas e diretas. A questão levantada é que o tempo que o sistema leva para realizar um treinamento ou uma atribuição é muito alto em relação ao tempo recomendado para uma resposta de requisição. Diante desse problema, a solução foi o uso de um serviço de e-mail que notifica o usuário quando a atribuição ou o treinamento estiver concluído. Por meio de bibliotecas do ecossistema Spring e do *ThymeLeaf* o usuário era notificado de forma eficiente quando o sistema terminasse de processar sua requisição.

Por fim, o uso do Amazon S3 dá-se pelo motivo de que todos os arquivos, sejam eles de texto, training-ppm ou fotos, eram armazenados localmente, podendo ocasionar problemas de memória ou segurança. A implementação de um serviço que liga o sistema ao Amazon S3 é necessária, portanto, para que esse problema seja resolvido. Utilizando bibliotecas próprias da Amazon em Java, foi possível realizar a integração do WhoAuthor com o Amazon S3, integração essa que permite o armazenamento de todos os arquivos utilizados dentro do sistema em um servidor remoto seguro, simples e acessível.

4.4.7 QUARTO ESTÁGIO DE DESENVOLVIMENTO: *FRONT-END*

Para melhorar a relação entre o usuário e o WhoAuthor, iniciou-se o desenvolvimento de um protótipo para o *front-end* do WhoAuthor utilizando a linguagem de programação TypeScript juntamente com a biblioteca React.ts. Apesar de inicialmente as telas se encontrarem estáticas, sem qualquer comunicação com o *back-end*, todas se encontram formuladas e arquitetadas dentro do ecossistema do React.ts.

A interface do front-end do WhoAuthor foi desenhada e montada utilizando o Figma, um editor gráfico online. Tal montagem levou em consideração conceitos de UI e

UX para o seu desenvolvimento.

4.5 RESULTADO DOS TESTES

Como explicado no tópico 5.1, o sistema é treinado em variações de parâmetros, logo, os testes realizados seguirão a mesma regra. Foram feitos testes com 24 autores clássicos da literatura brasileira, listados na tabela 2; os tamanhos dos arquivos submetidos foram de 48KB e 200KB; os resultados também são subdivididos nos contextos (quantidade de símbolos anteriores e posteriores considerados no momento de inferir a classificação gramatical de um termo); por fim, os testes ainda são divididos de acordo com as variações propostas pela validação cruzada, como mostra a tabela 1.

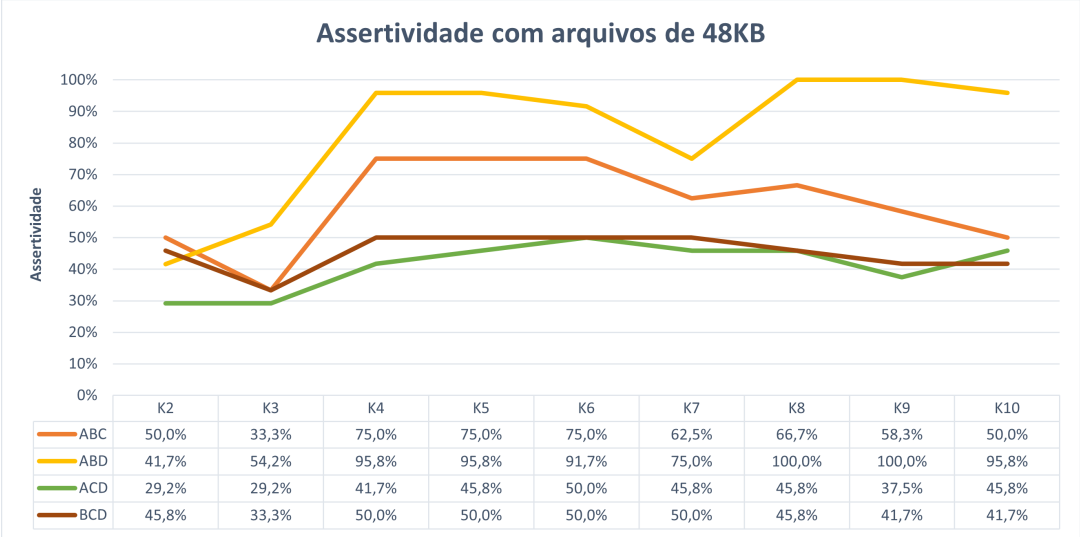
Tabela 2 – Autores presentes no *corpus* do sistema

Nome do autor(a)
Adolfo Caminha
Alcântara Machado
Aluísio Azevedo
Bernardo Guimarães
Camilo Castelo Branco
Clarice Lispector
Érico Veríssimo
Euclides Cunha
Fernando Sabino
Graciliano Ramos
Joaquim Manoel Macedo
Jorge Amado
José de Alencar
João Ubaldo
Júlia Almeida
Lima Barreto
Luís Fernando Veríssimo
Machado de Assis (Obras Realistas)
Machado de Assis (Obras Românticas)
Mario Prata
Monteiro Lobato
Raul Pompéia
Rubem Fonseca
Visconde Taunay

Ao longo das experimentações, realizou-se um total de 1728 testes, número esse que pode ser explicado da seguinte forma: todos os 24 autores foram testados, nas quatro variações de trecho para a validação cruzada, nos nove contextos, tendo sido todo esse processo feito duas vezes, para 48KB e para 200KB. Logo, a multiplicação desses números vai resultar em 1728. Os gráficos presentes nas figuras 17 e 18 seguintes ilustram a assertividade desses testes considerando as variáveis supracitadas:

A utilização de nove contextos diferentes foi feita para encontrar o ponto máximo da curva de aprendizado característica do PPM; quando ele é identificado, não é necessário

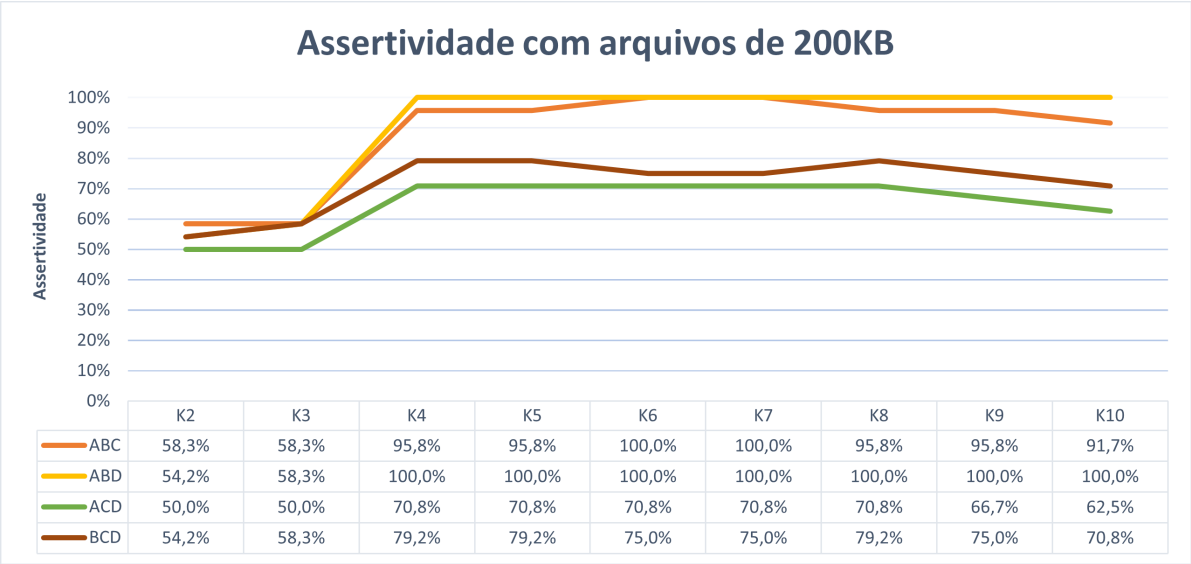
Figura 17 – Resultados com arquivos de 48KB



	K2	K3	K4	K5	K6	K7	K8	K9	K10
ABC	50,0%	33,3%	75,0%	75,0%	75,0%	62,5%	66,7%	58,3%	50,0%
ABD	41,7%	54,2%	95,8%	95,8%	91,7%	75,0%	100,0%	100,0%	95,8%
ACD	29,2%	29,2%	41,7%	45,8%	50,0%	45,8%	45,8%	37,5%	45,8%
BCD	45,8%	33,3%	50,0%	50,0%	50,0%	50,0%	45,8%	41,7%	41,7%

Fonte: Autoria própria

Figura 18 – Resultados com arquivos de 200KB



	K2	K3	K4	K5	K6	K7	K8	K9	K10
ABC	58,3%	58,3%	95,8%	95,8%	100,0%	100,0%	95,8%	95,8%	91,7%
ABD	54,2%	58,3%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%	100,0%
ACD	50,0%	50,0%	70,8%	70,8%	70,8%	70,8%	70,8%	66,7%	62,5%
BCD	54,2%	58,3%	79,2%	79,2%	75,0%	75,0%	79,2%	75,0%	70,8%

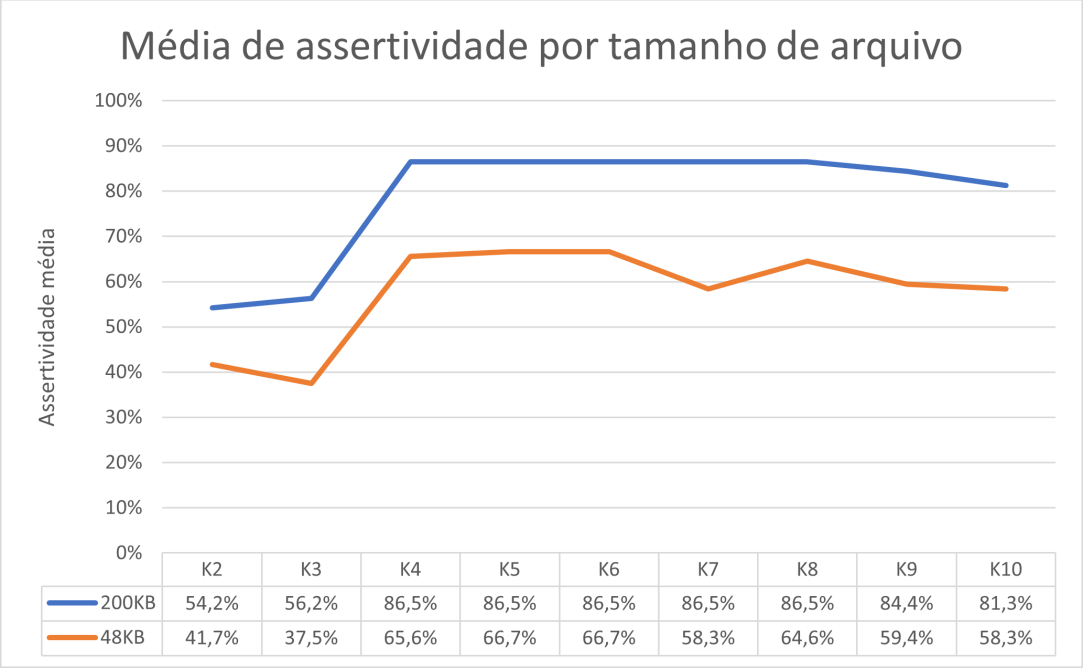
Fonte: Autoria própria

aumentar o número do contexto, pois o desempenho da compressão não vai melhorar, uma vez que atingiu seu ápice (BARUFALDI et al., 2010).

Com o objetivo de identificar as melhores condições de operação do sistema, foi feita uma média dos resultados das variações da validação cruzada para cada contexto, nos dois tamanhos de arquivo utilizados. O gráfico presente na figura 19 mostra as respectivas médias:

Analisando a linha azul (correspondente aos testes com arquivos de 200KB), é perceptível que o melhor resultado -86,5%- é atingido quando o contexto utilizado é igual

Figura 19 – Média das variações de cada validação



	K2	K3	K4	K5	K6	K7	K8	K9	K10
200KB	54,2%	56,2%	86,5%	86,5%	86,5%	86,5%	86,5%	84,4%	81,3%
48KB	41,7%	37,5%	65,6%	66,7%	66,7%	58,3%	64,6%	59,4%	58,3%

Fonte: Autoria própria

a quatro. A partir daí, os números se mantêm estáveis até ocorrer um decréscimo de 2,1 pontos percentuais quando K=9. Isso comprova que o ponto máximo da curva, quando o tamanho dos arquivos é de 200KB, é atingido no contexto quatro; então, dessa posição em diante, o resultado se manterá e, em seguida, diminuirá.

A observação da linha laranja (correspondente aos testes com arquivos de 48KB) mostra um aumento considerável no valor dos resultados do contexto três para o quatro; porém, é no contexto cinco que o ponto máximo da curva é alcançado. Nessa posição, a média das validações cruzadas é de 66,7%, valor que é mantido, também, no contexto seis; o decréscimo, por conseguinte, é percebido a partir do contexto sete. Nessa perspectiva, os experimentos realizados demonstraram que, sendo feita uma diminuição no tamanho do arquivo, foi necessário um contexto mais alto para atingir o ponto máximo da curva característica do PPM. Uma vez que a comprovação dessa hipótese não é o foco deste trabalho, mais testes devem ser realizados futuramente para mais investigações sobre o comportamento do algoritmo PPM-C. Esses gráficos apresentam os resultados de forma mais generalizada, conforme o detalhamento dos dados para cada autor descrito no Apêndice B.

Dos 24 autores eleitos para o os experimentos, nas melhores condições (arquivo de 200KB e contexto quatro), 13 obtiveram 100% de acerto nos testes, nove ficaram com o percentual de 75% de acerto e dois com 50% de acerto. Apesar de mais da metade dos literatos terem alcançado 100% de acerto, nove deles ficaram com um resultado favorável

(75%) e dois com uma porcentagem mais baixa (50%). Dadas as particularidades de cada autor e, ainda, variáveis como região, escola literária e período histórico, que influenciam na escrita dos indivíduos, é compreensível que nem todos tenham obtido os melhores resultados nas mesmas condições.

Os dois autores que obtiveram apenas 50% de acerto são representantes pioneiros do Romantismo Brasileiro (1836-1881); já entre os que obtiveram 75% ou 100% de acerto, são encontrados alguns românticos, mas a predominância é de autores pós-românticos. Em parte, esse fato justifica os resultados, pois, como já explicado, o Romantismo foi um marco para a mudança na forma como era compreendido o conceito de autoria. A partir dessa escola literária, os autores passaram a valorizar mais as características individuais em detrimento do modelo de escrita vigente. Nas escolas que sucederam a romântica, a subjetividade estilística na escrita é ainda mais valorizada: isso implica em autores com impressões digitais autorais cada vez mais distintas umas das outras – o que, por conseguinte, facilita o trabalho de classificação do WhoAuthor. É possível, ao menos parcialmente, atribuir à referida subjetividade os resultados obtidos na melhor condição, uma vez que autores mais recentes tiveram mais acertos nas classificações do que autores do início do Romantismo – época literária, conforme visto na introdução, na qual ainda desabrochava a compreensão de autoria como existe atualmente.

Vale ressaltar que esses testes contam com o acréscimo da pontuação durante o treinamento. Com o intuito de investigar a influência que teria a pontuação no processo de atribuição de autoria, o sistema foi implementado para considerar todos os sinais, como pontos, vírgulas etc. Após obtidos os resultados, foi feita uma comparação com a porcentagem de testes executados anteriormente - que foram aplicados na construção do Projeto Integrador. Considerando as mesmas condições, contexto quatro e arquivos 200KB, sem a consideração da pontuação, a média de acerto foi de 84,38%; com a pontuação, esse número subiu para 86,5%, um aumento de 2,12 pontos percentuais. Apesar de singelo, esse aumento confirma que levar em conta a pontuação influencia positivamente o processo de atribuição de autoria, reforçando a ideia de Zhao e Zobel (ZHAO; ZOBEL, 2007).

4.6 PROJETOS DE PESQUISA

No ano de 2019, após a formação da equipe, ficou clara a relevância do trabalho que seria realizado, bem como a sua amplitude – quantidade de tarefas, por exemplo – a fim de que fossem obtidos os presentes resultados. Assim, após identificado o caráter científico, o trabalho foi submetido e aceito nos editais para projeto de pesquisa do Instituto Federal do Rio Grande do Norte duas vezes. O primeiro projeto de pesquisa – intitulado de "Atribuição de Autoria de Textos Literários Baseado nas Classificações Gramaticais utilizando o PPM"– teve início em 11 de agosto de 2020 e foi concluído no dia 31 de julho de 2021, tendo sido aproveitados os seus resultados na construção do Projeto Integrador. No

entanto, tendo em vista que diversos aspectos no software ainda poderiam ser melhorados e mais testes realizados, em 10 de maio de 2021, foi principiado um novo projeto de pesquisa, que tem por título "Classificação e Atribuição de Autoria de Textos da Literatura Brasileira com a utilização da variação C do algoritmo de compressão Prediction by Partial Matching (PPM-C)". Desse projeto, cujo encerramento está previsto para dezembro de 2021, deriva-se o presente Trabalho de Conclusão de Curso.

4.7 SUBMISSÃO DE ARTIGO PARA O SBSI

Considerada a concordância deste trabalho com a bibliografia científica relacionada, os bons resultados de testes e a variedade de aplicações que pode ter o sistema, foi decidida a submissão de um artigo sobre o WhoAuthor para o Simpósio Brasileiro de Sistemas de Informação (SBSI). No momento da escrita do presente relatório, esse processo está em andamento e será concluído de acordo com os prazos de submissão definidos pelo SBSI.

4.8 REGISTRO DE SOFTWARE

A realização do registro de software foi considerada imprescindível desde o início dos trabalhos, porque, de acordo com o Instituto Nacional da Propriedade Industrial,

> o registro de programa de computador é fundamental para comprovar a autoria de seu desenvolvimento perante o Poder Judiciário, podendo ser muito útil em casos de processos relativos a concorrência desleal, cópias não autorizadas, pirataria, etc., garantindo, assim, maior segurança jurídica ao seu detentor para proteger o seu ativo de negócio. (INPI, 2020)

Por isso, após a finalização do primeiro projeto de pesquisa, foi realizado um registro do software, cujo número do processo é BR512021000120-6. No momento da escrita deste relatório, os trâmites para a efetivação de um segundo registro – dessa vez do WhoAuthor –, que apresenta todas as implementações supracitadas, estão em andamento.

5 CONCLUSÃO

A atribuição de autoria é uma atividade que teve sua importância ressignificada ao longo dos anos, ganhando, hodiernamente, não só maior relevância como também novas aplicações. Sob essa ótica, aliá-la à tecnologia da informação é a forma mais viável de atender a todas as demandas, diminuir a jornada de trabalho de profissionais de linguística que trabalham na área e possibilitar uma análise massiva de dados.

Dessa forma, o objetivo geral de desenvolver um software que, utilizando o PPM-C, pudesse reconhecer os padrões das sequências de classificações gramaticais e, assim, atribuir a autoria de textos brasileiros, foi efetivado. Além disso, foram feitas melhorias para facilitar o manuseio do usuário, como a automatização do processo de testes e treinamento, aprimoramento da interface, comodidade no treinamento e testagem do sistema com métodos de simples entendimento, utilização de serviço de e-mail, melhor comunicação com o usuário e diversas outras funcionalidades.

A realização de inúmeros testes e treinamentos, por sua vez, também possibilitou a definição dos melhores parâmetros para utilização do sistema (arquivos de tamanho maior, 200KB, e contexto igual a quatro) de forma a oferecer a maior quantidade possível de acertos (resultando em uma média de 86,5% de acertos). Esse percentual, quando comparado ao de trabalhos como o de Baayen (BAAYEN et al., 2002)- o qual, utilizando o LDA alcançou uma precisão entre 81,5% e 88,1%- mostra-se fortemente competitivo. Outrossim, os resultados do WhoAuthor podem ser equiparados aos de algoritmos bastante utilizados nessa área de pesquisa, como o SVM e o MDA que, no trabalho de Ebrahimpour (EBRAHIMPOUR et al., 2013), apresentaram mais de 90% de desempenho. Apesar desse valor ser superior à média de 86,5% do WhoAuthor, deve ser ponderado o fato de Ebrahimpour (EBRAHIMPOUR et al., 2013), em vários casos, ter usado uma quantidade de textos maior do que a utilizada neste trabalho. Mesmo que essa ação converta-se em resultados positivos sobre o desempenho do método, ela não é facilmente aplicada a todas as situações reais, já que um autor pode dispor de poucos textos para o treinamento do sistema. Nesse quesito, o WhoAuthor apresenta vantagens, pois mesmo com a reduzida quantidade de textos por autor, os resultados obtidos foram bastante favoráveis. A consideração dos sinais de pontuação também se mostrou uma decisão acertada, pois melhorou os resultados, contribuindo para o aumento da confiabilidade do sistema. Isso pode ser confirmado tanto pela concordância com a bibliografia relacionada, como pelo resultado dos testes realizados durante o Projeto Integrador que – nas mesmas condições, agora eleitas como ideais (contexto igual a quatro e tamanho de arquivo de 200KB) – foi de 84,38%, inferior em 2,12 pontos percentuais ao resultado dos testes em que a pontuação estava incluída (86,5%).

Por fim, considerando a influência de fatores como o curto tempo para realização do trabalho e as limitações impostas pela pandemia do Covid-19, alguns testes, implementações e encontros para capacitação foram retardados ou completamente anulados. Todavia, os resultados obtidos foram satisfatórios, e viabilizam o uso do sistema, pois como afirma Joula (JUOLA, 2006) a perfeição, em muitos casos, não é necessária para a utilidade do programa. Ademais desses fatores, é importante salientar o fato do trabalho englobar duas disciplinas bastante distintas: a linguística, uma ciência subjetiva, e a informática, uma ciência objetiva, Enquadrar todas as nuances de uma língua nos parâmetros de um sistema é uma tarefa cheia de limitações, já que a linguagem humana consiste em um sistema enlouquecedor de variabilidade com sutis regularidades (JUOLA, 2006).

5.1 TRABALHOS FUTUROS

A língua e a tecnologia são mutáveis e se transformam com o passar do tempo, o que não é diferente para os softwares. Por isso, adaptações para outras esferas de atuação e implementações podem ser examinadas e aplicadas em trabalhos futuros a fim de garantir a aplicabilidade e aumento da confiabilidade do sistema.

Dada a vastidão da área compreendida pela atribuição de autoria, o sistema pode atuar não só na esfera da pesquisa literária, como também com as devidas modificações, atender demandas judiciais ou empresariais. Logo, trabalhos futuros podem explorar essas possibilidades e o desempenho do sistema em cada uma delas.

Outrossim, testes com arquivos de mais variações de tamanho podem ser realizados, para investigar o comportamento do algoritmo diante das mais diversas situações em que o usuário pode se encontrar, com abundância ou escassez de textos de determinado autor.Além disso, também é válida a realização de testes no intuito de analisar o comportamento do algoritmo diante da relação entre o tamanho dos arquivos de texto e o número dos contextos.

Ademais, uma análise mais aprofundada dos elementos linguísticos de cada literato que compõem o *corpus* pode ser realizada, com o fito de justificar o fato de alguns deles não terem apresentado o melhor desempenho nas condições ideais eleitas.

Para mais, considerando as limitações que os usuários podem ter (um autor com poucos textos disponíveis, por exemplo) e a necessidade de processamento, podem ser feitas implementações no sistema, de forma geral, visando, sempre, atingir o máximo de confiabilidade, a exemplo de:

- Finalizar o front-end;
- Utilizar conceitos como containers e microservices para melhorar o processamento e a arquitetura do software;

- Criar sistema de seções e login para dar mais comordidade ao usuário, dando acesso a um históriooco de atribuições;
- Desenvolver uma área de treinamento somente para usuários administradores;
- Implementar a extração de textos de arquivos PDF.

REFERÊNCIAS

AMARAL, C. E. R. do. *Fake news é crime no Brasil*. 2020. Disponível em: <{https://jus.com.br/artigos/82580/fake-news-e-crime-no-brasi}.>

ARAUJO, R. M. de. EstilÍstica aplicada (a expressividade dos textos literÁrios). s.d.

AWS, A. *Recursos do Amazon S3 — Amazon Web Service*. [S.l.]: https://aws.amazon.com/pt/s3/features/Storage$_m$*anagement*$_a$*nd*$_m$*onitoring*, 2021.

BAAYEN, H. et al. An experiment in authorship attribution. In: CITESEER. *6th JADT*. [S.l.], 2002. v. 1, p. 69–75.

BARUFALDI, B. et al. Classificação automática de textos por período literário utilizando compressão de dados através do ppm-c. *Linguamática*, v. 2, n. 1, p. 35–43, 2010.

BAUER, M.; AARTS, B. A construção do corpus: Um princípio para coleta de dados qualitativos. pesquisa qualitativa com texto, imagem e som: um manual prático. *Martin W. Bauer, George Gaskell (ed.)*, p. 39–63, 2010.

BRANDAO, S. Atribuição de autoria: problema antigo, novas ferramentas. *Texto Digital*, v. 1, 06 2006.

BRASIL. *Constituição da República Federativa do Brasil: promulgada em 5 de outubro de 1988*. [S.l.: s.n.], 1988.

CHOCIAY, R. Em busca do estilo. *ALFA: Revista de Linguística*, v. 27, 1983.

CLEARY, J.; TEAHAN, W.; WITTEN, I. Unbounded length contexts for ppm. *Data Compression Conference Proceedings*, v. 40, 03 2003.

DEACON, J. Model-view-controller (mvc) architecture. *Online][Citado em: 10 de março de 2006.] http://www.jdl.co.uk/briefings/MVC.pdf*, 2009.

DING, Y. H.; LIU, C. H.; TANG, Y. X. Mvc pattern based on java. In: TRANS TECH PUBL. *Applied Mechanics and Materials*. [S.l.], 2012. v. 198, p. 537–541.

EBRAHIMPOUR, M. et al. Automated authorship attribution using advanced signal classification techniques. *PloS one*, Public Library of Science, v. 8, n. 2, p. e54998, 2013.

ECLIPSE. *Enabling Open Innovation Collaboration | The Eclipse Foundation*. 2021-03. ed. [S.l.]: https://www.eclipse.org, 2021.

FERREIRA, E. de P.; JR, M. M. *Aplicações RESTful*. 2018.

FIELDING, R. T. *Architectural styles and the design of network-based software architectures*. [S.l.]: University of California, Irvine, 2000.

FIGMA. *Figma*. [S.l.]: https://www.figma.com, 2021.

FILHO, B. Élisson Sampaio Cavalcante; Francisco Dantas Nobre Neto; Leonardo Vidal Batista; Glauco de Sousa e Silva; Ana Paula Nunes Guimarães; Elenilson Vieira da S. AtrubuiÇÃo de autoria de textos literÁrios utilizando ppm e svm. *62ª Reunião Anual da SBPC*, 2010.

FORBES. *12 países com maior exposição a fake news*. 2018. Disponível em: <{https://forbes.com.br/listas/2018/06/12-paises-com-maior-exposicao-a-fake-new}.>

GITHUB. *GitHub: Where the world builds software*. [s.n.], 2021. Disponível em: <{https://github.com}.>

HANSEN, J. A. *A sátira e o engenho: Gregório de Matos e a Bahia do século XVII*. [S.l.]: Ateliê Editorial, 2004.

HARVARD. *Harvard University*. c2019. Disponível em: <{https://www.harvard.edu}.>

INPI. *Programas de Computador*. 2020. Disponível em: <https://www.gov.br/inpi/pt-br/servicos/perguntas-frequentes/programas-de-computador#faq1.0>.

INSOMNIA. *The API Design Plataform and API Client - Insomnia*. 2021.5.3. ed. [S.l.]: https://insomnia.rest, 2021.

JAVA. *Java Standart Edition*. 14 lts. ed. [S.l.]: https://www.oracle.com/java/technologies/downloads/, 2021.

JAVASCRIPT. *Javascript | MDN*. [S.l.]: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript, 2021.

JOCKERS, M. L.; WITTEN, D. M. A comparative study of machine learning methods for authorship attribution. *Literary and Linguistic Computing*, Oxford University Press, v. 25, n. 2, p. 215–223, 2010.

JPA, S. *Spring Data JPA*. 2.5.5. ed. [S.l.]: https://spring.io/projects/spring-data-jpa, 2021.

JSON. *Introducing JSON*. [S.l.]: https://www.json.org/json-en.html, 2021.

JUOLA, P. Authorship attribution. *Information Retrieval*, v. 1, n. 3, p. 233–334, 2006.

LOELIGER, J.; MCCULLOUGH, M. *Version Control with Git: Powerful tools and techniques for collaborative software development*. [S.l.]: "O'Reilly Media, Inc.", 2012.

MAVEN. *Maven - Welcome to Apache Maven*. [s.n.], 2021. Disponível em: <{https://maven.apache.or}.>

MICHAELIS. *MICHAELIS moderno dicionário da língua portuguesa*. Disponível em: <{http://michaelis.uol.com.br/moderno/portugues/index.ph}.>

MICROSOFT. *Informações do parâmetro, listar membros e informações rápidas - Visual Studio (Windows) | Microsoft Docs*. 2019. Acesso em: 24 de julho. 2021. Disponível em: <{https://docs.microsoft.com/pt-br/visualstudio/ide/using-intellisense?view=vs-2019}.>

MICROSOFT. *TypeScript, The complete beginners guide*. [S.l.: s.n.], 2019.

MILANI, A. *PostgreSQL-Guia do Programador*. [S.l.]: Novatec Editora, 2008.

MURTY, J. *Programming amazon web services: S3, EC2, SQS, FPS, and SimpleDB*. [S.l.]: "O'Reilly Media, Inc.", 2008.

OAKS, S.; WONG, H. *Java threads*. [S.l.]: O'Reilly Media, Inc., 1999.

PINTO, F. B. B. A. A escrita fragmentada de sousândrade e a memória do trauma: reflexões. 2017.

POSTGRESQL. *PostgreSQL: The World's Most Advanced Open Source Relational Database*. 14.0. ed. [S.l.]: https://www.postgresql.org, 2021.

PRESS, C. U. *Cambridge Dictionary - make your words meaningful*. ONLINE, 2021. Disponível em: <https://dictionary.cambridge.org>.

PRIBERAM. *Dicionário Priberam Online de Português Contemporâneo*. 2021. Acesso em: 05 de Agosto. 2020. Disponível em: <{https://dicionario.priberam.org}.>

REACTTS. *TypeScript: Documentation - React*. [s.n.], 2021. Disponível em: <{https://www.typescriptlang.org/docs/handbook/react.htm}.>

ROCKETSEAT. *TypeScript: Documentation - React*. 2021. Acesso em: 24 de julho. 2021. Disponível em: <{https://blog.rocketseat.com.br/typescript-vantagens-mitos-conceitos}. >

RONCARI, L. *Literatura brasileira: dos primeiros cronistas aos últimos românticos*. [S.l.]: Edusp, 1995. v. 2.

SABBAGH, R. *Scrum: Gestão ágil para projetos de sucesso*. [S.l.]: Editora Casa do Código, 2014.

SANTIAGO, A. *Mesmo sendo crime, casos de plágios ainda fazem parte do mundo acadêmico*. 2018. Disponível em: <{https://bityli.com/iR2AM}.>

SPRING. *Spring Overview*. 2.26. ed. [S.l.]: https://spring.io, 2021.

SPRINGTOOLS. *Spring Tools 4 for Eclipse*. 4. ed. [S.l.]: https://spring.io/tools, 2021.

STACKSHARE. *Insomnia REST Client*. 2016. Acesso em: 24 de julho. 2021. Disponível em: <{https://stackshare.io/insomnia-rest-clien}.>

STAMATATOS, E. Authorship attribution using text distortion. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. [S.l.: s.n.], 2017. p. 1138–1149.

VARANASI, B. *Introducing Maven: A Build Tool for Today's Java Developers*. [S.l.]: Apress, 2019.

VSCODE. *Visual Studio Code - Code Editing. Redefined*. 1.6. ed. [S.l.]: https://code.visualstudio.com, 2021.

ZHAO, Y.; ZOBEL, J. Searching with style: Authorship attribution in classic literature. In: CITESEER. *ACM International Conference Proceeding Series*. [S.l.], 2007. v. 244, p. 59–68.

APÊNDICE

A VARIABILIDADE DE TREINAMENTO

Livros usados no treinamento	Livro usados nos testes	Contexto	Tamanho unificado
Texto A, Texto B, Texto C	Texto D	2	48KB
Texto A, Texto B, Texto D	Texto C	2	48KB
Texto A, Texto C, Texto D	Texto B	2	48KB
Texto B, Texto C e Texto D	Texto A	2	48KB
Texto A, Texto B, Texto C	Texto D	2	200BK
Texto A, Texto B, Texto D	Texto C	2	200KB
Texto A, Texto C, Texto D	Texto B	2	200KB
Texto B, Texto C e Texto D	Texto A	2	200KB
Texto A, Texto B, Texto C	Texto D	3	48KB
Texto A, Texto B, Texto D	Texto C	3	48KB
Texto A, Texto C, Texto D	Texto B	3	48KB
Texto B, Texto C e Texto D	Texto A	3	48KB
Texto A, Texto B, Texto C	Texto D	3	200BK
Texto A, Texto B, Texto D	Texto C	3	200KB
Texto A, Texto C, Texto D	Texto B	3	200KB
Texto B, Texto C e Texto D	Texto A	3	200KB
Texto A, Texto B, Texto C	Texto D	4	48KB
Texto A, Texto B, Texto D	Texto C	4	48KB
Texto A, Texto C, Texto D	Texto B	4	48KB
Texto B, Texto C e Texto D	Texto A	4	48KB
Texto A, Texto B, Texto C	Texto D	4	200BK
Texto A, Texto B, Texto D	Texto C	4	200KB
Texto A, Texto C, Texto D	Texto B	4	200KB
Texto B, Texto C e Texto D	Texto A	4	200KB
Texto A, Texto B, Texto C	Texto D	5	48KB
Texto A, Texto B, Texto D	Texto C	5	48KB
Texto A, Texto C, Texto D	Texto B	5	48KB
Texto B, Texto C e Texto D	Texto A	5	48KB
Texto A, Texto B, Texto C	Texto D	5	200BK
Texto A, Texto B, Texto D	Texto C	5	200KB

Livros usados no treinamento	Livro usados nos testes	Contexto	Tamanho unificado
Texto A, Texto C, Texto D	Texto B	5	200KB
Texto B, Texto C e Texto D	Texto A	5	200KB
Texto A, Texto B, Texto C	Texto D	6	48KB
Texto A, Texto B, Texto D	Texto C	6	48KB
Texto A, Texto C, Texto D	Texto B	6	48KB
Texto B, Texto C e Texto D	Texto A	6	48KB
Texto A, Texto B, Texto C	Texto D	6	200BK
Texto A, Texto B, Texto D	Texto C	6	200KB
Texto A, Texto C, Texto D	Texto B	6	200KB
Texto B, Texto C e Texto D	Texto A	6	200KB
Texto A, Texto B, Texto C	Texto D	7	48KB
Texto A, Texto B, Texto D	Texto C	7	48KB
Texto A, Texto C, Texto D	Texto B	7	48KB
Texto B, Texto C e Texto D	Texto A	7	48KB
Texto A, Texto B, Texto C	Texto D	7	200BK
Texto A, Texto B, Texto D	Texto C	7	200KB
Texto A, Texto C, Texto D	Texto B	7	200KB
Texto B, Texto C e Texto D	Texto A	7	200KB
Texto A, Texto B, Texto C	Texto D	8	48KB
Texto A, Texto B, Texto D	Texto C	8	48KB
Texto A, Texto C, Texto D	Texto B	8	48KB
Texto B, Texto C e Texto D	Texto A	8	48KB
Texto A, Texto B, Texto C	Texto D	8	200BK
Texto A, Texto B, Texto D	Texto C	8	200KB
Texto A, Texto C, Texto D	Texto B	8	200KB
Texto B, Texto C e Texto D	Texto A	8	200KB
Texto A, Texto B, Texto C	Texto D	9	48KB
Texto A, Texto B, Texto D	Texto C	9	48KB
Texto A, Texto C, Texto D	Texto B	9	48KB
Texto B, Texto C e Texto D	Texto A	9	48KB
Texto A, Texto B, Texto C	Texto D	9	200BK
Texto A, Texto B, Texto D	Texto C	9	200KB

Livros usados no treinamento	Livro usados nos testes	Contexto	Tamanho unificado
Texto A, Texto C, Texto D	Texto B	9	200KB
Texto B, Texto C e Texto D	Texto A	9	200KB
Texto A, Texto B, Texto C	Texto D	10	48KB
Texto A, Texto B, Texto D	Texto C	10	48KB
Texto A, Texto C, Texto D	Texto B	10	48KB
Texto B, Texto C e Texto D	Texto A	10	48KB
Texto A, Texto B, Texto C	Texto D	10	200BK
Texto A, Texto B, Texto D	Texto C	10	200KB
Texto A, Texto C, Texto D	Texto B	10	200KB
Texto B, Texto C e Texto D	Texto A	10	200KB

B RESULTADOS POR AUTOR

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Adolfo Caminha	48KB	K2	25,00%
Adolfo Caminha	200KB	K2	25,00%
Adolfo Caminha	48KB	K3	25,00%
Adolfo Caminha	200KB	K3	25,00%
Adolfo Caminha	48KB	K4	50,00%
Adolfo Caminha	200KB	K4	75,00%
Adolfo Caminha	48KB	K5	50,00%
Adolfo Caminha	200KB	K5	75,00%
Adolfo Caminha	48KB	K6	50,00%
Adolfo Caminha	200KB	K6	75,00%
Adolfo Caminha	48KB	K7	50,00%
Adolfo Caminha	200KB	K7	75,00%
Adolfo Caminha	48KB	K8	50,00%
Adolfo Caminha	200KB	K8	100,00%
Adolfo Caminha	48KB	K9	50,00%
Adolfo Caminha	200KB	K9	75,00%
Adolfo Caminha	48KB	K10	50,00%
Adolfo Caminha	200KB	K10	50,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Alcântara Machado	48KB	K2	75,00%
Alcântara Machado	200KB	K2	50,00%
Alcântara Machado	48KB	K3	75,00%
Alcântara Machado	200KB	K3	50,00%
Alcântara Machado	48KB	K4	75,00%
Alcântara Machado	200KB	K4	75,00%
Alcântara Machado	48KB	K5	75,00%
Alcântara Machado	200KB	K5	75,00%
Alcântara Machado	48KB	K6	75,00%
Alcântara Machado	200KB	K6	75,00%
Alcântara Machado	48KB	K7	75,00%
Alcântara Machado	200KB	K7	75,00%
Alcântara Machado	48KB	K8	75,00%
Alcântara Machado	200KB	K8	75,00%
Alcântara Machado	48KB	K9	75,00%
Alcântara Machado	200KB	K9	75,00%
Alcântara Machado	48KB	K10	75,00%
Alcântara Machado	200KB	K10	75,00%
Aluísio Azevedo	48KB	K2	75,00%
Aluísio Azevedo	200KB	K2	75,00%
Aluísio Azevedo	48KB	K3	75,00%
Aluísio Azevedo	200KB	K3	75,00%
Aluísio Azevedo	48KB	K4	100,00%
Aluísio Azevedo	200KB	K4	100,00%
Aluísio Azevedo	48KB	K5	75,00%
Aluísio Azevedo	200KB	K5	100,00%
Aluísio Azevedo	48KB	K6	100,00%
Aluísio Azevedo	200KB	K6	100,00%
Aluísio Azevedo	48KB	K7	75,00%
Aluísio Azevedo	200KB	K7	100,00%
Aluísio Azevedo	48KB	K8	100,00%
Aluísio Azevedo	200KB	K8	100,00%
Aluísio Azevedo	48KB	K9	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Aluísio Azevedo	200KB	K9	100,00%
Aluísio Azevedo	48KB	K10	75,00%
Aluísio Azevedo	200KB	K10	100,00%
Bernardo Guimarães	48KB	K2	75,00%
Bernardo Guimarães	200KB	K2	75,00%
Bernardo Guimarães	48KB	K3	75,00%
Bernardo Guimarães	200KB	K3	75,00%
Bernardo Guimarães	48KB	K4	100,00%
Bernardo Guimarães	200KB	K4	100,00%
Bernardo Guimarães	48KB	K5	100,00%
Bernardo Guimarães	200KB	K5	100,00%
Bernardo Guimarães	48KB	K6	100,00%
Bernardo Guimarães	200KB	K6	100,00%
Bernardo Guimarães	48KB	K7	100,00%
Bernardo Guimarães	200KB	K7	100,00%
Bernardo Guimarães	48KB	K8	75,00%
Bernardo Guimarães	200KB	K8	100,00%
Bernardo Guimarães	48KB	K9	75,00%
Bernardo Guimarães	200KB	K9	100,00%
Bernardo Guimarães	48KB	K10	75,00%
Bernardo Guimarães	200KB	K10	100,00%
Camilo Castelo Branco	48KB	K2	50,00%
Camilo Castelo Branco	200KB	K2	75,00%
Camilo Castelo Branco	48KB	K3	0,00%
Camilo Castelo Branco	200KB	K3	100,00%
Camilo Castelo Branco	48KB	K4	75,00%
Camilo Castelo Branco	200KB	K4	100,00%
Camilo Castelo Branco	48KB	K5	75,00%
Camilo Castelo Branco	200KB	K5	100,00%
Camilo Castelo Branco	48KB	K6	75,00%
Camilo Castelo Branco	200KB	K6	100,00%
Camilo Castelo Branco	48KB	K7	50,00%
Camilo Castelo Branco	200KB	K7	100,00%
Camilo Castelo Branco	48KB	K8	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Camilo Castelo Branco	200KB	K8	75,00%
Camilo Castelo Branco	48KB	K9	75,00%
Camilo Castelo Branco	200KB	K9	75,00%
Camilo Castelo Branco	48KB	K10	75,00%
Camilo Castelo Branco	200KB	K10	75,00%
Clarice Lispector	48KB	K2	75,00%
Clarice Lispector	200KB	K2	50,00%
Clarice Lispector	48KB	K3	75,00%
Clarice Lispector	200KB	K3	50,00%
Clarice Lispector	48KB	K4	75,00%
Clarice Lispector	200KB	K4	100,00%
Clarice Lispector	48KB	K5	75,00%
Clarice Lispector	200KB	K5	100,00%
Clarice Lispector	48KB	K6	75,00%
Clarice Lispector	200KB	K6	100,00%
Clarice Lispector	48KB	K7	75,00%
Clarice Lispector	200KB	K7	100,00%
Clarice Lispector	48KB	K8	75,00%
Clarice Lispector	200KB	K8	100,00%
Clarice Lispector	48KB	K9	75,00%
Clarice Lispector	200KB	K9	100,00%
Clarice Lispector	48KB	K10	75,00%
Clarice Lispector	200KB	K10	100,00%
Érico Veríssimo	48KB	K2	50,00%
Érico Veríssimo	200KB	K2	75,00%
Érico Veríssimo	48KB	K3	75,00%
Érico Veríssimo	200KB	K3	75,00%
Érico Veríssimo	48KB	K4	75,00%
Érico Veríssimo	200KB	K4	75,00%
Érico Veríssimo	48KB	K5	75,00%
Érico Veríssimo	200KB	K5	75,00%
Érico Veríssimo	48KB	K6	75,00%
Érico Veríssimo	200KB	K6	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Érico Veríssimo	48KB	K7	50,00%
Érico Veríssimo	200KB	K7	75,00%
Érico Veríssimo	48KB	K8	75,00%
Érico Veríssimo	200KB	K8	75,00%
Érico Veríssimo	48KB	K9	75,00%
Érico Veríssimo	200KB	K9	75,00%
Érico Veríssimo	48KB	K10	75,00%
Érico Veríssimo	200KB	K10	75,00%
Euclides Cunha	48KB	K2	50,00%
Euclides Cunha	200KB	K2	75,00%
Euclides Cunha	48KB	K3	25,00%
Euclides Cunha	200KB	K3	75,00%
Euclides Cunha	48KB	K4	50,00%
Euclides Cunha	200KB	K4	100,00%
Euclides Cunha	48KB	K5	50,00%
Euclides Cunha	200KB	K5	100,00%
Euclides Cunha	48KB	K6	50,00%
Euclides Cunha	200KB	K6	100,00%
Euclides Cunha	48KB	K7	50,00%
Euclides Cunha	200KB	K7	100,00%
Euclides Cunha	48KB	K8	75,00%
Euclides Cunha	200KB	K8	100,00%
Euclides Cunha	48KB	K9	75,00%
Euclides Cunha	200KB	K9	100,00%
Euclides Cunha	48KB	K10	75,00%
Euclides Cunha	200KB	K10	75,00%
Fernando Sabino	48KB	K2	50,00%
Fernando Sabino	200KB	K2	50,00%
Fernando Sabino	48KB	K3	25,00%
Fernando Sabino	200KB	K3	50,00%
Fernando Sabino	48KB	K4	50,00%
Fernando Sabino	200KB	K4	75,00%
Fernando Sabino	48KB	K5	50,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Fernando Sabino	200KB	K5	75,00%
Fernando Sabino	48KB	K6	50,00%
Fernando Sabino	200KB	K6	75,00%
Fernando Sabino	48KB	K7	25,00%
Fernando Sabino	200KB	K7	75,00%
Fernando Sabino	48KB	K8	50,00%
Fernando Sabino	200KB	K8	75,00%
Fernando Sabino	48KB	K9	50,00%
Fernando Sabino	200KB	K9	75,00%
Fernando Sabino	48KB	K10	50,00%
Fernando Sabino	200KB	K10	75,00%
Graciliano Ramos	48KB	K2	75,00%
Graciliano Ramos	200KB	K2	100,00%
Graciliano Ramos	48KB	K3	50,00%
Graciliano Ramos	200KB	K3	100,00%
Graciliano Ramos	48KB	K4	75,00%
Graciliano Ramos	200KB	K4	100,00%
Graciliano Ramos	48KB	K5	75,00%
Graciliano Ramos	200KB	K5	100,00%
Graciliano Ramos	48KB	K6	75,00%
Graciliano Ramos	200KB	K6	100,00%
Graciliano Ramos	48KB	K7	75,00%
Graciliano Ramos	200KB	K7	100,00%
Graciliano Ramos	48KB	K8	50,00%
Graciliano Ramos	200KB	K8	100,00%
Graciliano Ramos	48KB	K9	25,00%
Graciliano Ramos	200KB	K9	75,00%
Graciliano Ramos	48KB	K10	25,00%
Graciliano Ramos	200KB	K10	75,00%
Joaquim Manoel Macedo	48KB	K2	25,00%
Joaquim Manoel Macedo	200KB	K2	25,00%
Joaquim Manoel Macedo	48KB	K3	25,00%
Joaquim Manoel Macedo	200KB	K3	25,00%
Joaquim Manoel Macedo	48KB	K4	50,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Joaquim Manoel Macedo	200KB	K4	50,00%
Joaquim Manoel Macedo	48KB	K5	50,00%
Joaquim Manoel Macedo	200KB	K5	50,00%
Joaquim Manoel Macedo	48KB	K6	50,00%
Joaquim Manoel Macedo	200KB	K6	50,00%
Joaquim Manoel Macedo	48KB	K7	50,00%
Joaquim Manoel Macedo	200KB	K7	50,00%
Joaquim Manoel Macedo	48KB	K8	50,00%
Joaquim Manoel Macedo	200KB	K8	50,00%
Joaquim Manoel Macedo	48KB	K9	25,00%
Joaquim Manoel Macedo	200KB	K9	50,00%
Joaquim Manoel Macedo	48KB	K10	50,00%
Joaquim Manoel Macedo	200KB	K10	50,00%
Jorge Amado	48KB	K2	25,00%
Jorge Amado	200KB	K2	0,00%
Jorge Amado	48KB	K3	0,00%
Jorge Amado	200KB	K3	0,00%
Jorge Amado	48KB	K4	75,00%
Jorge Amado	200KB	K4	100,00%
Jorge Amado	48KB	K5	75,00%
Jorge Amado	200KB	K5	100,00%
Jorge Amado	48KB	K6	75,00%
Jorge Amado	200KB	K6	100,00%
Jorge Amado	48KB	K7	75,00%
Jorge Amado	200KB	K7	100,00%
Jorge Amado	48KB	K8	75,00%
Jorge Amado	200KB	K8	75,00%
Jorge Amado	48KB	K9	75,00%
Jorge Amado	200KB	K9	75,00%
Jorge Amado	48KB	K10	75,00%
Jorge Amado	200KB	K10	75,00%
José de Alencar	48KB	K2	75,00%
José de Alencar	200KB	K2	75,00%
José de Alencar	48KB	K3	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
José de Alencar	200KB	K3	75,00%
José de Alencar	48KB	K4	50,00%
José de Alencar	200KB	K4	50,00%
José de Alencar	48KB	K5	50,00%
José de Alencar	200KB	K5	50,00%
José de Alencar	48KB	K6	50,00%
José de Alencar	200KB	K6	50,00%
José de Alencar	48KB	K7	50,00%
José de Alencar	200KB	K7	50,00%
José de Alencar	48KB	K8	50,00%
José de Alencar	200KB	K8	50,00%
José de Alencar	48KB	K9	25,00%
José de Alencar	200KB	K9	50,00%
José de Alencar	48KB	K10	25,00%
José de Alencar	200KB	K10	50,00%
João Ubaldo	48KB	K2	50,00%
João Ubaldo	200KB	K2	100,00%
João Ubaldo	48KB	K3	75,00%
João Ubaldo	200KB	K3	75,00%
João Ubaldo	48KB	K4	75,00%
João Ubaldo	200KB	K4	100,00%
João Ubaldo	48KB	K5	75,00%
João Ubaldo	200KB	K5	100,00%
João Ubaldo	48KB	K6	75,00%
João Ubaldo	200KB	K6	100,00%
João Ubaldo	48KB	K7	75,00%
João Ubaldo	200KB	K7	100,00%
João Ubaldo	48KB	K8	50,00%
João Ubaldo	200KB	K8	100,00%
João Ubaldo	48KB	K9	50,00%
João Ubaldo	200KB	K9	100,00%
João Ubaldo	48KB	K10	50,00%
João Ubaldo	200KB	K10	100,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Júlia Almeida	48KB	K2	25,00%
Júlia Almeida	200KB	K2	0,00%
Júlia Almeida	48KB	K3	25,00%
Júlia Almeida	200KB	K3	0,00%
Júlia Almeida	48KB	K4	25,00%
Júlia Almeida	200KB	K4	75,00%
Júlia Almeida	48KB	K5	25,00%
Júlia Almeida	200KB	K5	75,00%
Júlia Almeida	48KB	K6	25,00%
Júlia Almeida	200KB	K6	75,00%
Júlia Almeida	48KB	K7	25,00%
Júlia Almeida	200KB	K7	75,00%
Júlia Almeida	48KB	K8	25,00%
Júlia Almeida	200KB	K8	100,00%
Júlia Almeida	48KB	K9	25,00%
Júlia Almeida	200KB	K9	100,00%
Júlia Almeida	48KB	K10	25,00%
Júlia Almeida	200KB	K10	100,00%
Lima Barreto	48KB	K2	25,00%
Lima Barreto	200KB	K2	25,00%
Lima Barreto	48KB	K3	0,00%
Lima Barreto	200KB	K3	75,00%
Lima Barreto	48KB	K4	75,00%
Lima Barreto	200KB	K4	75,00%
Lima Barreto	48KB	K5	75,00%
Lima Barreto	200KB	K5	75,00%
Lima Barreto	48KB	K6	75,00%
Lima Barreto	200KB	K6	75,00%
Lima Barreto	48KB	K7	50,00%
Lima Barreto	200KB	K7	75,00%
Lima Barreto	48KB	K8	75,00%
Lima Barreto	200KB	K8	75,00%
Lima Barreto	48KB	K9	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Lima Barreto	200KB	K9	75,00%
Lima Barreto	48KB	K10	75,00%
Lima Barreto	200KB	K10	75,00%
Luís Fernando Veríssimo	48KB	K2	0,00%
Luís Fernando Veríssimo	200KB	K2	25,00%
Luís Fernando Veríssimo	48KB	K3	25,00%
Luís Fernando Veríssimo	200KB	K3	0,00%
Luís Fernando Veríssimo	48KB	K4	75,00%
Luís Fernando Veríssimo	200KB	K4	75,00%
Luís Fernando Veríssimo	48KB	K5	75,00%
Luís Fernando Veríssimo	200KB	K5	75,00%
Luís Fernando Veríssimo	48KB	K6	75,00%
Luís Fernando Veríssimo	200KB	K6	75,00%
Luís Fernando Veríssimo	48KB	K7	75,00%
Luís Fernando Veríssimo	200KB	K7	75,00%
Luís Fernando Veríssimo	48KB	K8	75,00%
Luís Fernando Veríssimo	200KB	K8	75,00%
Luís Fernando Veríssimo	48KB	K9	75,00%
Luís Fernando Veríssimo	200KB	K9	75,00%
Luís Fernando Veríssimo	48KB	K10	75,00%
Luís Fernando Veríssimo	200KB	K10	75,00%
Machado de Assis (Obras Realistas)	48KB	K2	25,00%
Machado de Assis (Obras Realistas)	200KB	K2	25,00%
Machado de Assis (Obras Realistas)	48KB	K3	25,00%
Machado de Assis (Obras Realistas)	200KB	K3	25,00%
Machado de Assis (Obras Realistas)	48KB	K4	25,00%
Machado de Assis (Obras Realistas)	200KB	K4	75,00%
Machado de Assis (Obras Realistas)	48KB	K5	50,00%
Machado de Assis (Obras Realistas)	200KB	K5	75,00%
Machado de Assis (Obras Realistas)	48KB	K6	50,00%
Machado de Assis (Obras Realistas)	200KB	K6	75,00%
Machado de Assis (Obras Realistas)	48KB	K7	50,00%
Machado de Assis (Obras Realistas)	200KB	K7	75,00%
Machado de Assis (Obras Realistas)	48KB	K8	50,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Machado de Assis (Obras Realistas)	200KB	K8	100,00%
Machado de Assis (Obras Realistas)	48KB	K9	50,00%
Machado de Assis (Obras Realistas)	200KB	K9	100,00%
Machado de Assis (Obras Realistas)	48KB	K10	50,00%
Machado de Assis (Obras Realistas)	200KB	K10	100,00%
Machado de Assis (Obras Românticas)	48KB	K2	25,00%
Machado de Assis (Obras Românticas)	200KB	K2	50,00%
Machado de Assis (Obras Românticas)	48KB	K3	25,00%
Machado de Assis (Obras Românticas)	200KB	K3	75,00%
Machado de Assis (Obras Românticas)	48KB	K4	50,00%
Machado de Assis (Obras Românticas)	200KB	K4	100,00%
Machado de Assis (Obras Românticas)	48KB	K5	50,00%
Machado de Assis (Obras Românticas)	200KB	K5	100,00%
Machado de Assis (Obras Românticas)	48KB	K6	50,00%
Machado de Assis (Obras Românticas)	200KB	K6	100,00%
Machado de Assis (Obras Românticas)	48KB	K7	50,00%
Machado de Assis (Obras Românticas)	200KB	K7	100,00%
Machado de Assis (Obras Românticas)	48KB	K8	50,00%
Machado de Assis (Obras Românticas)	200KB	K8	100,00%
Machado de Assis (Obras Românticas)	48KB	K9	50,00%
Machado de Assis (Obras Românticas)	200KB	K9	100,00%
Machado de Assis (Obras Românticas)	48KB	K10	50,00%
Machado de Assis (Obras Românticas)	200KB	K10	100,00%
Mario Prata	48KB	K2	25,00%
Mario Prata	200KB	K2	100,00%
Mario Prata	48KB	K3	0,00%
Mario Prata	200KB	K3	100,00%
Mario Prata	48KB	K4	75,00%
Mario Prata	200KB	K4	100,00%
Mario Prata	48KB	K5	100,00%
Mario Prata	200KB	K5	100,00%
Mario Prata	48KB	K6	100,00%
Mario Prata	200KB	K6	100,00%
Mario Prata	48KB	K7	100,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Mario Prata	200KB	K7	100,00%
Mario Prata	48KB	K8	100,00%
Mario Prata	200KB	K8	100,00%
Mario Prata	48KB	K9	75,00%
Mario Prata	200KB	K9	100,00%
Mario Prata	48KB	K10	75,00%
Mario Prata	200KB	K10	100,00%
Monteiro Lobato	48KB	K2	0,00%
Monteiro Lobato	200KB	K2	25,00%
Monteiro Lobato	48KB	K3	25,00%
Monteiro Lobato	200KB	K3	25,00%
Monteiro Lobato	48KB	K4	50,00%
Monteiro Lobato	200KB	K4	75,00%
Monteiro Lobato	48KB	K5	50,00%
Monteiro Lobato	200KB	K5	75,00%
Monteiro Lobato	48KB	K6	25,00%
Monteiro Lobato	200KB	K6	75,00%
Monteiro Lobato	48KB	K7	25,00%
Monteiro Lobato	200KB	K7	75,00%
Monteiro Lobato	48KB	K8	25,00%
Monteiro Lobato	200KB	K8	75,00%
Monteiro Lobato	48KB	K9	25,00%
Monteiro Lobato	200KB	K9	75,00%
Monteiro Lobato	48KB	K10	25,00%
Monteiro Lobato	200KB	K10	75,00%
Raul Pompéia	48KB	K2	25,00%
Raul Pompéia	200KB	K2	50,00%
Raul Pompéia	48KB	K3	25,00%
Raul Pompéia	200KB	K3	50,00%
Raul Pompéia	48KB	K4	75,00%
Raul Pompéia	200KB	K4	100,00%
Raul Pompéia	48KB	K5	75,00%
Raul Pompéia	200KB	K5	100,00%
Raul Pompéia	48KB	K6	75,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Raul Pompéia	200KB	K6	100,00%
Raul Pompéia	48KB	K7	50,00%
Raul Pompéia	200KB	K7	100,00%
Raul Pompéia	48KB	K8	75,00%
Raul Pompéia	200KB	K8	75,00%
Raul Pompéia	48KB	K9	75,00%
Raul Pompéia	200KB	K9	75,00%
Raul Pompéia	48KB	K10	50,00%
Raul Pompéia	200KB	K10	75,00%
Rubem Fonseca	48KB	K2	75,00%
Rubem Fonseca	200KB	K2	100,00%
Rubem Fonseca	48KB	K3	75,00%
Rubem Fonseca	200KB	K3	75,00%
Rubem Fonseca	48KB	K4	100,00%
Rubem Fonseca	200KB	K4	100,00%
Rubem Fonseca	48KB	K5	100,00%
Rubem Fonseca	200KB	K5	100,00%
Rubem Fonseca	48KB	K6	100,00%
Rubem Fonseca	200KB	K6	100,00%
Rubem Fonseca	48KB	K7	50,00%
Rubem Fonseca	200KB	K7	100,00%
Rubem Fonseca	48KB	K8	75,00%
Rubem Fonseca	200KB	K8	100,00%
Rubem Fonseca	48KB	K9	75,00%
Rubem Fonseca	200KB	K9	100,00%
Rubem Fonseca	48KB	K10	50,00%
Rubem Fonseca	200KB	K10	75,00%
Visconde Taunay	48KB	K2	0,00%
Visconde Taunay	200KB	K2	50,00%
Visconde Taunay	48KB	K3	0,00%
Visconde Taunay	200KB	K3	75,00%
Visconde Taunay	48KB	K4	50,00%
Visconde Taunay	200KB	K4	100,00%
Visconde Taunay	48KB	K5	50,00%

Nome do autor	Contexto	Tamanho de arquivo	Assertividade
Visconde Taunay	200KB	K5	100,00%
Visconde Taunay	48KB	K6	50,00%
Visconde Taunay	200KB	K6	100,00%
Visconde Taunay	48KB	K7	50,00%
Visconde Taunay	200KB	K7	100,00%
Visconde Taunay	48KB	K8	75,00%
Visconde Taunay	200KB	K8	100,00%
Visconde Taunay	48KB	K9	75,00%
Visconde Taunay	200KB	K9	100,00%
Visconde Taunay	48KB	K10	75,00%
Visconde Taunay	200KB	K10	100,00%