UNIVERSIDADE DE CAXIAS DO SUL

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Ricardo Dacol Gil

Desenvolvimento de um Sistema de Análise de Semântica Latente para Avaliar Produções Textuais

Caxias do Sul, RS

2016

Ricardo Dacol Gil

Desenvolvimento de um Sistema de Análise de Semântica Latente para Avaliar Produções Textuais

Trabalho de Conclusão de Curso para obtenção do Grau de Bacharel em Ciência da Computação da Universidade de Caxias do Sul

Orientadora: Carine Geltrudes Webber

Caxias do Sul, RS

2016

Agradecimentos

Agradeço aos meus pais, Gelson e Elaine, pela paciência, carinho, e apoio que sempre me deram.

Aos meus irmãos, Fernando e Juliana, por liderarem por exemplo.

A minha namorada, Juliana, pela ajuda nos momentos de dúvida, e incentivo nos momentos preguiça.

A minha orientadora, Carine, pela ajuda e atenção, pela paciência demonstrada e a cobrança pelos resultados.

RESUMO

Este trabalho apresenta um estudo sobre Processamento de Linguagem Natural e Mineração de Texto, abordando os principais conceitos e técnicas, como a classificação de textos e sumarização. No contexto educacional, existem muitas produções textuais que são manualmente analisadas pelos professores. A necessidade de desenvolvimento de softwares que podem automatizar os processos de leitura e revisão de textos se mostra aparente. Estes softwares auxiliam a percepção dos professores no processo de ensino e acompanhamento dos alunos. Também pode apoiar o próprio aluno na compreensão de textos. Alguns trabalhos e softwares relacionados já foram desenvolvidos, como a ferramenta Análise de Produções Textuais (LOVATO, 2015), que será utilizada neste estudo. Além de minerar os textos e apresentá-los de forma organizada, é necessário aplicar técnicas avançadas de análise para fornecer resultados mais refinados. Neste enfoque, é proposto o desenvolvimento do Analisador de Semântica Latente, capaz de realizar o treinamento de determinada área do conhecimento e efetuar a mineração de produções textuais de estudantes para fins de acompanhamento da evolução dos mesmos, oferecendo um conjunto de ferramentas funcionais para a análise de textos educacionais.

Palavras-chave: mineração de textos educacionais. análise de semântica latente. ferramentas educacionais.

ABSTRACT

This project presents a study about Text Mining, exploring the main concepts and techniques, such as text classification and summarization. In the context of education, there are many textual productions that must be manually analyzed by the professors. The need to develop software capable of automating the process of text reading and analysis is apparent. These softwares help the professor's perception of his or her students in the teaching process and monitoring of the students. Some existing works and software have already been developed, such as the Textual Production Analyzer (LOVATO, 2015) which will be used in this study. Apart from mining texts and presenting information in an organized fashion, one must apply advanced techniques to produce more refined results. In this scope, it is proposed to develop a Latent Semantic Analyzer, capable of learning certain aspects of knowledge and mining student's textual productions to in order to better accompany their educational evolution, offering functional tools for the analysis of educational texts.

Key words: educational text mining. latent semantic analysis. educational tools.

Lista de ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

LSA	Análise de Semântica Latente
MT	Mineração de Texto
PLN	Processamento de Linguagem Natural
SVM	Máquina de Suporte Vetorial
SVD	Decomposição em Valores Singulares
VI	Visualização de Informação

SUMÁRIO

1 Introdução

Com a crescente popularização da internet e o número cada vez maior de usuários, frequentemente surge a necessidade de automatizar tarefas diárias, tornando as tarefas do cotidiano procedimentos automáticos. Isto se mostra evidente na área de educação, onde se tem um grande volume de informação disponível, em sua maioria na forma de textos (SILVA, 2004). A partir disso, torna-se necessário eliminar informação que o usuário considera desnecessária, mantendo apenas o conteúdo interessante, fornecendo resultados de forma a auxiliar o aprendizado.

Segundo Rodrigues, Ramos, Silva e Gomes (2013), já não se discute a importância do uso das tecnologias digitais da informação e comunicação na área de educação. Sendo assim, a Mineração de Texto (MT) pode contribuir com a área de informática na educação. O objetivo da MT é identificar palavras, termos e expressões relevantes que se destacam nos textos processados.

Existe uma área conhecida como Descoberta de Conhecimento, que se refere ao processamento de informações em textos e se divide em dois grupos: de dados estruturados, que envolve a mineração de dados, usualmente em bancos de dados; e de dados não estruturados, que trata do processamento de documentos textuais. Pode-se dizer que a MT é uma subárea da mineração de dados que trata do processamento de dados não estruturados, sendo a maioria em formato de textos, conforme demonstrado na Figura 1.

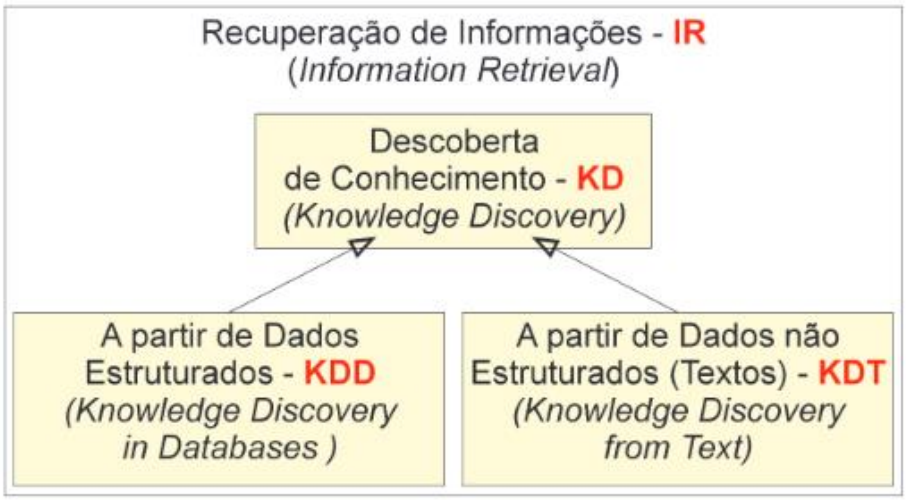


Figura 1: Descoberta de Conhecimento (MORAIS; AMBRÓSIO, 2007)

Nesta direção, diversos algoritmos baseados em processamento de linguagem natural já foram desenvolvidos e testados para mineração de texto. A maioria deles foi desenvolvida para funcionar em língua inglesa. Reutilizar algoritmos que processam outro

idioma nem sempre produz bons resultados. Contudo, pode-se generalizar a solução e substituir o vocabulário por termos em língua portuguesa. Neste caso, diversos trabalhos apontam para a aplicação do algoritmo de máquinas de suporte vetorial. Outra ferramenta analítica existente para interpretar o sentido descrito pelas palavras extraídas, e por consequência, dos documentos analisados, é a Análise de Semântica Latente (LSA). Ela propõe a criação de um mapeamento de palavras e documentos em um espaço comum, computado através das frequências de palavras, ou do logaritmo de frequências de palavras (DELL, 2015) (EMMS; LUZ, 2011).

Neste trabalho é proposto o desenvolvimento de um algoritmo que realiza a mineração em artigos científicos, seguindo os processos de MT e embasado nas metodologias de LSA que serão aprofundados nos capítulos a seguir. O funcionamento do sistema se dará em duas etapas: na primeira será feito o treinamento do sistema para que este seja capaz de definir as palavras de maior significado nos textos de treinamento. Na segunda etapa, será avaliada a produção textual dos alunos contra os textos de treinamento, visando atribuir uma nota ao aluno quanto à qualidade de sua produção textual.

1.1 Objetivos Geral e Específicos

Este trabalho propõe implementar um algoritmo baseado na técnica de análise de semântica latente para incluir a funcionalidade de extração de significados de textos. Ele será aplicado na análise de produções textuais de estudantes que constituirão a amostra do estudo. O resultado deste trabalho será inserido no contexto do projeto Undermine Text Miner (desenvolvido na UCS). O objetivo do projeto é desenvolver uma ferramenta de visualização de dados que permita analisar e avaliar produções textuais de alunos de graduação.

Para atingir o objetivo geral, este trabalho compreende os seguintes objetivos específicos:

A) realizar pesquisa sobre o estado da arte das técnicas de análise de textos baseados em semântica latente.
B) escolher um algoritmo dentro da área de análise de semântica latente a ser implementado.
C) implementar o algoritmo de análise de semântica latente.
D) testar e coletar resultados produzidos pelo algoritmo.
E) comparar os resultados produzidos pelo algoritmo com um especialista humano (possivelmente um professor de uma disciplina cursada pelos estudantes alvo da amostra).
F) integrar o programa desenvolvido ao software Undermine Text Miner
G) realizar testes de integração do programa com o Undermine Text Miner

1.2 Estrutura do Trabalho

Este trabalho está estruturado em quatro capítulos. O segundo capítulo trata da conceituação da mineração de texto, apresentando diversas aplicações e técnicas existentes. O terceiro capítulo aborda o conceito de Análise de Semântica Latente, que é a técnica escolhida para ser desenvolvida neste trabalho. O quarto capítulo detalha o funcionamento do software desenvolvido, abordando detalhes de seu funcionamento e resultados gerados. O quinto capítulo apresenta as conclusões finais sobre a implementação e testes realizados no trabalho.

2 Mineração de Texto

A mineração de texto (MT) pode ser definida como a análise textual de documentos com o objetivo de extrair informação ou conhecimento. Pode também ser aplicada para descobrir tendências que os documentos possam expor, para obter uma melhor visão sobre as pessoas, lugares e coisas fundamentados no que os documentos podem revelar. Ela pode classificar, organizar ou categorizar os documentos ou a informação que eles contêm, e sumarizar um documento em uma forma mais compacta. A MT pode funcionar com conjuntos de dados desestruturados ou semi-estruturados (EBECKEN; LOPES; COSTA, 2003).

A análise de texto na MT envolve a recuperação de informações, análise léxica a fim de estudar a frequência de distribuição de palavras, reconhecimento de padrões, identificação e anotação, extração de informações, técnicas de mineração de dados que incluem link e associação de análises, visualização e analítica preditiva. O objetivo maior é transformar o texto em dados para análise, por meio da aplicação do processamento de linguagem natural (PLN) e de métodos analíticos.

Informações importantes são obtidas normalmente pela elaboração de padrões e tendências através de meios como o padrão estatístico de aprendizagem. Geralmente a MT envolve o processo de estruturação do texto de entrada, de derivação de padrões dentro da estrutura de dados e, por fim, de avaliação e interpretação do resultado. O grau de importância em MT refere-se à combinação de relevância, originalidade e interesse. Tarefas típicas de MT incluem categorização, agrupamento de texto, extração de conceito, produção de taxonomias, análise de sentimentos, resumo de documentos e modelagem de relações entre entidades (SANGER; FELDMAN, 2006).

Em suma, a MT permite que os principais conceitos abordados em um texto sejam extraídos e relacionados, e o minerador apresenta então um grafo que possibilita a visualização do conteúdo de forma gráfica. Os benefícios da MT podem se estender a qualquer domínio que utilize textos, sendo que suas principais contribuições estão relacionadas à busca de informações específicas em documentos, à análise qualitativa e quantitativa de grandes volumes de textos, e a melhor compreensão do conteúdo disponível em documentos textuais (LOH, 2001).

2.1 Etapas de Mineração de Texto

Segundo Aranha (2007) um processo de MT é composto de cinco etapas (Figura 2). A primeira delas é a coleta de dados para formar uma base de dados textuais. A aquisição dos documentos (dados) ocorre neste momento. Os dados podem ser coletados de locais como pastas do disco dos computadores, discos móveis e bancos de dados. Pode também ser extraído da Internet, que dispõe de inúmeras fontes de textos, tais como sites, blogs e redes sociais como *Facebook* e *Twitter*. A obtenção desses textos e documentos provenientes da

internet é feita via robôs denominados *crawlers*, que navegam de forma automática, vasculhando todos os sites que se encontram disponíveis.

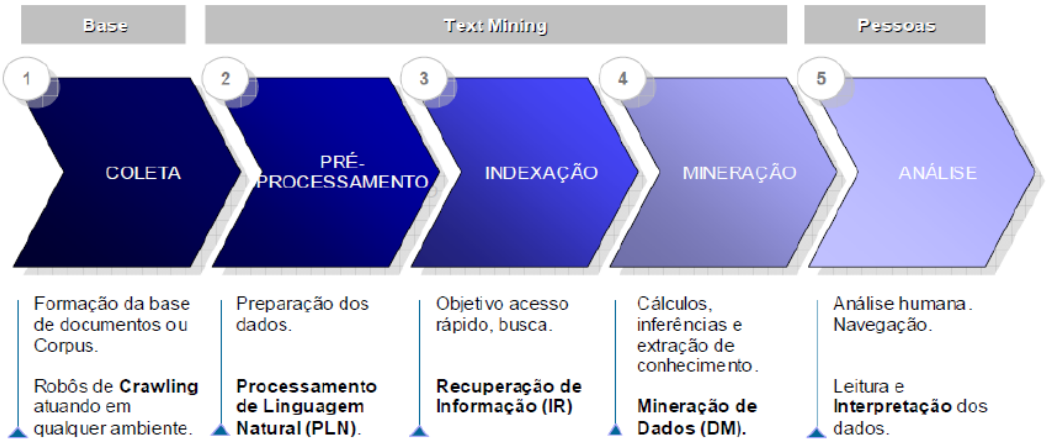


Figura 2: Processo de Mineração de Textos (ARANHA, 2007)

A segunda etapa, pré-processamento, busca estruturar os documentos, organizando-os, formatando-os e, consequentemente, melhorando sua qualidade para as etapas seguintes. É nesta etapa em que se removem os *StopWords*, se divide o texto em palavras, e classifica-as de acordo com a classe gramatical. Orengo e Huyck (2001) demonstram o processo de *stemming* que ocorre nesta etapa (Figura 3), onde as palavras são reduzidas às suas radicais.

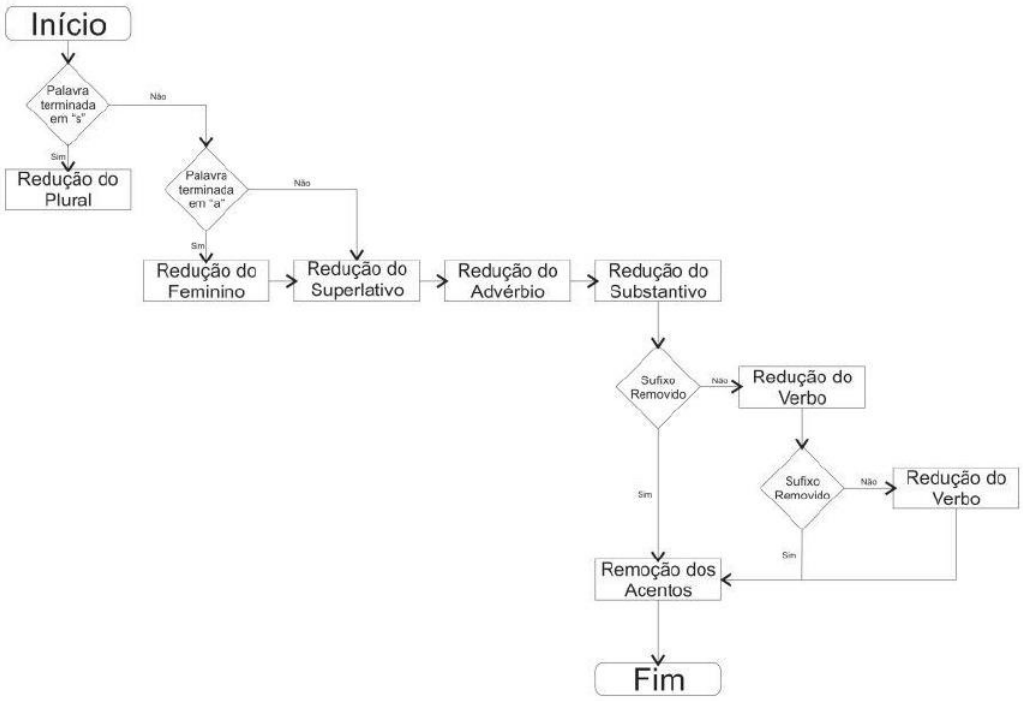


Figura 3: Algoritmo de *stemming* para a língua portuguesa (ORENGO; HUYCK, 2001)

As etapas para o processo de *stemming* são as seguintes:

1) Remoção de plural – Consiste em remover o "s" do final da palavra. No entanto, há uma lista de exceções, como a palavra "lápis", por exemplo.
2) Remoção de feminino – As palavras femininas são transformadas na correspondente masculina. Exemplo: "brasileira" será transformada em "brasileiro".
3) Remoção de advérbio – Como o único sufixo que identifica um advérbio é o sufixo "mente", o mesmo será retirado das palavras. Para este passo também há uma lista de exceções.
4) Remoção de aumentativo e diminutivo – Remove os sufixos dos substantivos e adjetivos que identificam aumentativo e diminutivo. Exemplo: "bonitinho" e "paredão".
5) Remoção de sufixos em nomes – Existe uma lista com 61 sufixos para substantivos e adjetivos. O programa testa a palavra contra esta lista, caso exista o sufixo é removido e as etapas seis e sete não são executadas.
6) Remoção de sufixos em verbos – Os verbos podem variar de acordo com o tempo, a pessoa, o número e o modo. A estrutura das formas verbais pode ser representada por: radical + vogal temática + tempo + pessoa. A finalidade deste passo é reduzir as formas verbais ao seu radical correspondente.
7) Remoção de vogais – Nesta etapa é removida a última vogal ("a", "e" ou "o") das palavras que não foram examinadas pelas etapas cinco e seis.
8) Remoção de acentos – Na última etapa, apenas a acentuação é retirada.

Os termos são organizados na base, de forma a facilitar seu acesso e recuperação na etapa de indexação.

Ao iniciar a etapa de pré-processamento, os termos podem ser analisados de duas maneiras: por meio da análise semântica ou por meio da análise estatística.

A análise semântica procura identificar a função que determinados termos exercem no texto (CORDEIRO, 2005). Fundamentada em técnicas de Processamento de Linguagem Natural (GLYMOUR *et al.*, 1997), essa análise demanda conhecimentos morfológico, semântico, sintático, pragmático, do discurso e do mundo, buscando denotar importância a um termo de acordo com sua contextualização textual (MORAIS; AMBRÓSIO, 2007).

Já a análise estatística determina a importância de um termo baseado na frequência com que aparece no documento. Considera-se, portanto, que a análise estatística visa estabelecer a frequência de cada termo ao longo do texto sem se importar com a contextualização do termo no texto (MORAIS; AMBRÓSIO, 2007).

A seguir, na etapa de mineração, é realizada a aplicação de algoritmos sobre os documentos com o intuito de extrair conhecimento desses textos conforme os objetivos desejados. Por exemplo, se a aplicação está voltada à área de educação, define-se a relevância dos termos e o objetivo que se deseja atingir na área. Algumas palavras possuem maior relevância no texto, portanto é necessário atribuir um valor aos termos, que pode ser baseado

na frequência com que o termo aparece no texto. As frequências mais comuns são: frequência absoluta, frequência relativa e frequência inversa de documentos.

A frequência absoluta representa a contagem de todas as vezes que um termo aparece num documento. A frequência relativa é a frequência absoluta dividida pelo tamanho do documento, ou seja, frequência absoluta de um termo dividida pela quantidade total de termos do documento. A frequência inversa de documentos utiliza o cálculo das duas frequências anteriores para ser calculada. Assim, é possível diminuir a importância de termos que aparecem muitas vezes e aumentar a importância de termos que aparecem poucas vezes, pois estes geralmente tem a propriedade de descrever melhor o assunto tratado (ARANHA, 2007).

Na última etapa é feita a análise e interpretação dos resultados obtidos. Essa análise não é feita por ferramentas computacionais, mas sim pelos usuários especialistas na área. É o especialista que irá verificar a compatibilidade dos resultados obtidos com o conhecimento disponível e o usuário irá validar a aplicabilidade dos resultados (ARANHA, 2007).

2.2 Aplicações de Mineração de Texto

A MT encontra diversas aplicações nos dias de hoje. Neste trabalho são tratadas as seguintes principais aplicações: Extração de informação, Rastreamento de Tópicos, Categorização, Agrupamento, Ligação de conceitos, Visualização de Informação, Resposta de Perguntas, Mineração de Regras de Associação, entre outros.

2.2.1 Extração de Informação

O processo de extração de informação (EI) identifica frases chave e relacionamentos dentro do texto e os extrai convertendo-os para uma estrutura tabular, com o objetivo de resumir o conteúdo do tema abordado no documento de forma legível. Além disso, ele infere as relações entre todas as pessoas identificadas, lugares, e hora para fornecer ao usuário a informação mais significativa. Essa tecnologia é útil quando se trata de uma grande quantidade de textos.

A mineração de dados tradicional assume que a informação se encontra numa base de dados relacional. No entanto, muitas vezes a informação digital só se encontra disponível no formato de documentos em linguagem natural, onde as técnicas tradicionais visam uma completa análise do texto. A EI por sua vez analisa apenas partes específicas do texto onde se encontram as informações relevantes, trazendo vantagens como menor complexidade de implementação e menor esforço computacional (EIKVIL, 1999).

2.2.2 Rastreamento de Tópicos

Rastreamento de tópicos é uma técnica que visa encontrar documentos relacionados a determinado tópico. Um sistema de rastreamento de tópicos cria um perfil de usuário baseado nos documentos que este visualiza, fornecendo assim outros documentos relacionados que a ele possam interessar. Existem ferramentas de rastreamento de tópicos que permitem ao usuário escolher palavras-chave e notificam o surgimento de documentos ou notícias relevantes às palavras escolhidas.

No entanto, essas ferramentas possuem uma limitação: no caso de o usuário utilizar a frase "Mineração de Texto", o mesmo poderá receber informações sobre mineração industrial, e não a respeito do conteúdo desejado. Algumas aplicações mais avançadas permitem escolher a que categorias pertencem as palavras-chave, e até pesquisar o histórico de navegação do usuário para inferir automaticamente as categorias.

Além disso, essa técnica tem um grande potencial de exploração comercial e científico. Comercialmente, empresas poderiam empregá-la para monitorar um concorrente na mídia, por exemplo. No âmbito científico, as publicações de artigos sobre tratamentos de uma determinada doença poderiam ser pré-selecionadas pela técnica e oferecidas ao usuário (GUPTA; LEHAL, 2009).

2.2.3 Sumarização

A sumarização automática vem sendo desenvolvida desde a década de 50, com o surgimento dos primeiros métodos para a produção de resumos, sendo o método das palavras-chave o mais significativo. Entretanto, os métodos "cegos", fazendo uso de técnicas superficiais, apresentavam inúmeros problemas nos resultados, tais como de coesão e de coerência, razão pela qual a área ficou estagnada nas décadas seguintes, voltando a despertar interesse com o advento da Internet e o aumento considerável de documentos disponíveis on-line. Mesmo que os computadores modernos consigam identificar pessoas, lugares e tempo, o desafio principal está na capacidade do software de analisar a semântica e interpretar o significado do texto (CHUANG; YANG, 2000).

Um dos pontos mais problemáticos, e, portanto sujeito a controvérsias, é discernir o que é relevante e o que pode ser descartado para compor um sumário. Afinal, a importância de uma frase ou trecho de um texto pode depender de vários fatores como os objetivos do autor do sumário, os objetivos ou interesse de seus possíveis leitores e a importância que o próprio autor, ou leitor, atribui às informações textuais.

Em contraposição, a abordagem profunda toma do processamento humano da linguagem a base interpretativa para a compreensão do texto-fonte e posterior produção do sumário correspondente. A base dessa metodologia é que um bom sistema automático de PLN faça uso de regras gramaticais e habilidades de inferência lógica, e manipule bases de

conhecimentos gerais, além do próprio conhecimento linguístico. A sumarização automática é baseada em teorias linguísticas formais. Assim, o sistema computacional deveria simular a inteligência humana, a fim de obter um processamento eficiente da língua. No entanto, para muitos pesquisadores, a tarefa de simular a inteligência humana para um domínio aberto, no PLN, ainda está fora do alcance.

Baseados em tal argumento, alguns pesquisadores utilizam as técnicas estatísticas como métodos superficiais, que seriam de mais simples aplicação, não necessitando de algoritmos complexos. Elas, porém não levam em consideração todo o conhecimento potencial da língua. Embora no passado parecesse necessário escolher entre essas duas abordagens, passou-se a buscar uma combinação coerente das mesmas, investigando amplamente tecnologias híbridas, ou seja, fazendo uso de metodologias simbólicas e técnicas estatísticas.

Quaisquer que sejam as metodologias e o foco da sumarização automática, os critérios de avaliação passaram a ocupar papel de destaque, tanto os de desempenho dos sumarizadores automáticos, quanto os da qualidade dos sumários produzidos. Tornaram-se frequentes as pesquisas sobre sumarização de documentos, cujo objetivo é a produção automática de sumários, analisando diversos textos-fonte que discorrem sobre o mesmo assunto (MARTINS et al., 2001).

2.2.4 Categorização

Categorização é o mesmo que a classificação automática de documentos em relação a um conjunto de categorias ou matérias predefinidas. Sendo assim, a categorização é responsável pela identificação dos temas principais de um documento, colocando-o num conjunto predefinido de tópicos. Durante o processamento, o software trata o texto como um ‘saco de palavras’. Assim, a categorização apenas calcula a frequência das palavras que estão contidas no texto, e a partir da contagem identifica os tópicos aos quais o documento está relacionado.

A categorização depende de um vocabulário para o qual os tópicos são predefinidos e as relações são identificadas através de termos gerais, termos específicos, sinônimos e termos relacionados. As ferramentas de categorização costumam ter técnicas para classificar os documentos que possuem mais conteúdo a respeito de um determinado tópico dentro de uma categoria (CORREA; LUDERMIR, 2002).

2.2.5 Agrupamento

Agrupamento é uma técnica utilizada para agrupar documentos similares entre si, porém diferencia-se da categorização por agrupar documentos sem utilização de tópicos predefinidos. Outra vantagem do agrupamento é que os textos podem pertencer a diversos subgrupos (Figura 4), garantindo que um documento relevante não será descartado de um resultado de pesquisa.

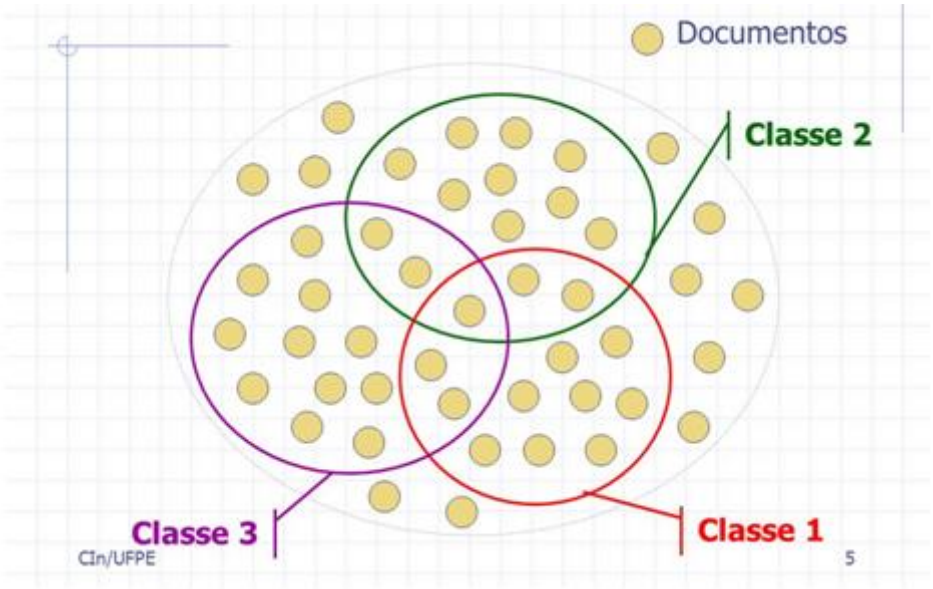


Figura 4: Documentos em diversos agrupamentos Fonte: http://slideplayer.com.br/slide/49276/

Um algoritmo básico gera um vetor de tópicos para cada documento e determina o peso do mesmo em relação a cada tópico. Essa tecnologia pode ser útil na organização de sistemas de gerenciamento de informação que podem conter milhares de fontes (STEINBACH; KARYPIS; KUMAR, 2000).

2.2.6 Ligação de Conceitos

Ferramentas de ligação de conceitos (LC) relacionam documentos identificando temas em comum, ajudando assim usuários a encontrar informações que não teriam encontrado através de métodos de busca convencionais. A LC promove a navegação em busca de informação e não a pesquisa direta por ela. A LC é valorizada na MT, principalmente na área de bioquímica onde a quantidade de pesquisas preexistentes torna impossível para os pesquisadores a leitura de todo o material para organizar pesquisas novas. O software de LC pode, por exemplo, identificar relações entre doenças e tratamentos onde humanos não conseguem (SHEHATA; KARRAY; KAMEL, 2010).

2.2.7 Visualização de Informação

Mineração de texto visual, ou visualização da informação (VI), coloca grandes fontes textuais numa hierarquia visual e fornece a capacidade de navegação, além de buscas simples, permitindo ao usuário analisar visualmente o conteúdo dos textos. A VI permite interagir com os documentos através da utilização de zoom, escalonamento, criação de submapas, entre outras técnicas que permitem reduzir o campo de documentos e explorar tópicos relacionados (WONG; WHITNEY; THOMAS, 1999).

2.2.8 Modelo “Resposta a Perguntas”

Outra aplicação do PLN é o processamento de perguntas, que lida com a identificação das melhores respostas a uma determinada pergunta. Diversos sites na internet são equipados com ferramentas de resposta a perguntas, permitindo ao usuário fazer uma pergunta ao computador e receber uma resposta.

As respostas são encontradas utilizando outras técnicas de MT como a extração de informação para identificar entidades como pessoas, lugares ou eventos; categorização para classificar a pergunta num tipo conhecido, como ‘quem’, ‘onde’, ’quando’, ’como’, etc., pois o sistema de pergunta-resposta baseia-se na ideia de que ambas são expressas utilizando o mesmo conjunto de palavras (JUÁREZ-GONZÁLEZ et al., 2006).

2.2.9 Mineração de Regras de Associação

A mineração de regras de associação tem como objetivo descobrir relações ou padrões frequentes entre variáveis num conjunto de dados. É aplicada em diversos cenários industriais e disciplinas, mas pouco utilizada nas ciências sociais, como educação. Dada uma base de registros, a MRA determina quais combinações de valores ocorrem com frequência, similar à análise de correlação em que relacionamentos entre duas variáveis são descobertos.

A tarefa de associação consiste em identificar quais atributos estão relacionados. Apresentam o seguinte formato: SE atributo X ENTÃO atributo Y. É uma das tarefas mais conhecidas devido aos bons resultados obtidos, principalmente nas análises de "Carrinhos de Compras" em sites de *e-commerce*, onde pode-se identificar quais produtos são levados juntos pelos consumidores. Outros exemplos seriam determinar os casos onde um novo medicamento pode apresentar efeitos colaterais, e identificar os usuários de planos que respondem positivamente a ofertas de novos serviços (VASCONCELOS; CARVALHO, 2004).

2.3 Pesquisas Relacionadas

A sumarização automática de documentos tem sido investigada ativamente na área do Processamento de Linguagem Natural (PLN) desde a segunda metade do século passado. Esta é atualmente uma área de investigação muito vasta, com inúmeros trabalhos publicados.

A sumarização automática abstrativa, que normalmente envolve a fusão da informação extraída, compressão e reformulação de frases, coloca uma forte ênfase na forma, com o objetivo de produzir sumários gramaticamente corretos, o que geralmente requer técnicas avançadas de geração de linguagem natural (ALMEIDA; MARTINS, 2013). Os sumários puramente extrativos são mais viáveis computacionalmente, e estas abordagens mais simples tornaram-se o padrão no campo da sumarização automática de textos.

Com a crescente popularização do uso de técnicas de aprendizagem automática em tarefas de PLN, surgiram publicações envolvendo abordagens estatísticas para a produção automática de sumários. Kupiec et al. (1995) descrevem um método que era capaz de aprender a partir de dados anotados manualmente. Uma função de classificação, baseada na abordagem naïve-Bayes, pode ser usada para categorizar cada frase como de interesse para o sumário ou não.

As propostas baseadas em aprendizagem automática apresentam geralmente a desvantagem de necessitarem de dados de treino sob a forma de frases extraídas desde documentos de texto (i.e., os exemplos de treino consistem de frases anotadas como interessantes ou não de pertencer a um sumário extrativo). Como forma de contornar a necessidade de obtenção de dados de treino, vários trabalhos anteriores exploraram ainda técnicas não-supervisionadas, por exemplo baseadas em fatoração de matrizes. Intuitivamente, estes métodos tentam agrupar as frases de um documento em grupos/componentes que sejam coerentes entre si, escolhendo posteriormente as frases mais representativas de cada grupo, de forma a construir os sumários.

Gong & Liu (2001) propuseram um método que utiliza LSA para selecionar frases adequadas à construção de um sumário. Este método cria inicialmente uma matriz de termos→⇥ frases, onde cada coluna representa o vetor de frequências ponderadas dos termos de uma frase. Após, é aplicada uma SVD sob a matriz, de forma a derivar a estrutura semântica latente. As frases com maiores pesos no primeiro conceito latente são finalmente selecionadas para a formação do sumário.

2.4 Técnicas e Algoritmos

Existem técnicas diferentes utilizadas na MT, das quais podemos citar o K-Nearest Neighbor (KNN), a Máquina de Suporte Vetorial (SVM), Classificador Bayesiano, e LSA.

2.4.1 K Nearest Neighbor

O algoritmo KNN é uma técnica clássica muito utilizada na MT. Para encontrar a palavra procurada, um classificador KNN é executado. Este é um tipo de aprendizagem baseada em instância, ou aprendizagem preguiçosa, onde a função é aproximada apenas localmente, e toda computação é deferida até a classificação.

Este método estima a distância entre duas frases para comparar e classificar o texto baseado na distância, onde X e Y representam as instâncias de dados e d é a distância entre X e Y. As maiores vantagens deste algoritmo são a precisão da classificação e a simplicidade de implementação. A de.svantagem é a grande utilização de memória da máquina onde é executado, e o tempo de execução (BIJALWAN *et al.*, 2014).

2.4.2 Máquina de Suporte Vetorial

Segundo Tan (2004), máquinas de suporte vetorial são modelos de aprendizagem supervisionados com algoritmos de aprendizagem associados que analisam dados e reconhecem padrões. São utilizados em classificadores e analisadores por regressão. Dado um conjunto de exemplos de treinamento marcados como pertencendo a uma determinada categoria, um algoritmo SVM constrói um modelo que indica a que categoria um novo exemplo se enquadra, o que torna o algoritmo um classificador linear binário não probabilístico.

Um modelo SVM é a representação dos exemplos como pontos num espaço, mapeados para que os exemplares de cada categoria sejam separados por uma faixa mais larga possível. Novos dados então são mapeados no mesmo espaço e são previstos para pertencer a uma categoria baseado em qual lado da faixa pertencerem.

Quando os dados não possuem rótulos, o aprendizado supervisionado não é possível, portanto um aprendizado não supervisionado é utilizado, o que faz com que o algoritmo obtenha o agrupamento natural dos dados, mapeando dados novos nestes agrupamentos. Este algoritmo de agrupamento é utilizado em aplicações industriais onde os dados não possuem rótulos, ou onde apenas alguns dados foram rotulados por um pré-processamento.

2.4.3 Classificador Bayesiano

Classificadores Bayesianos são classificadores estatísticos que classificam dados numa determinada classe baseando-se na probabilidade de pertencer a tal classe. O algoritmo fornece resultados rápidos e de grande correção quando aplicados a um volume grande de dados, sendo comparável aos resultados produzidos por redes neurais e árvores de decisão.

Os classificadores bayesianos simples supõem como hipótese de trabalho que o efeito do valor de um atributo não-classe é independente dos valores dos outros atributos. Ou seja, o valor de um atributo não afeta o valor dos outros atributos. Isto tem por objetivo facilitar o processamento envolvido na classificação (ZEMBRZUSKI, 2010).

2.4.4 Análise de Semântica Latente

A análise semântica latente (LSA) é um método frequentemente utilizado na área do PLN para analisar as relações entre documentos e os termos neles contidos, por meio da produção de um conjunto de conceitos latentes relacionados com os documentos e os termos. Este método assume que as palavras semanticamente relacionadas tendem a coocorrer nos textos. No contexto da aplicação em sumarização automática, uma matriz esparsa A representando as ocorrências de termos por frases, por exemplo, uma matriz com *m* linhas que representam os termos únicos, e com *n* colunas que representam cada frase, é construída a partir de um documento original.

Em seguida, é calculada uma decomposição em valores singulares (SVD) com o objetivo de reduzir o número de linhas, preservando a estrutura de similaridade entre as colunas. A SVD é tipicamente calculada com base num algoritmo que envolve duas etapas, o qual começa por reduzir a matriz A a sua forma bidiagonal, ou seja, uma matriz com entradas diferentes de zero apenas na diagonal principal e nos valores que se encontram acima ou abaixo, e que em seguida calcula a decomposição SVD da matriz bidiagonal através de um método iterativo.

Este tipo de análise emprega técnicas que avaliam a sequência dos termos no contexto dos textos, no sentido de identificar qual a sua função. Sua utilização justifica-se principalmente pela melhoria da qualidade dos resultados do processo de mineração de textos, especialmente se for incrementado por Processamento Linguístico.

A técnica de Análise de Semântica Latente explora a relação existente entre termos e os textos nos quais eles aparecem para construir um espaço vetorial onde similaridades de significado podem ser estabelecidas. Este novo espaço é conhecido como Espaço-Conceito ou Espaço Semântico (DEERWESTER, 1990), sendo que a proximidade entre significados é proporcional ao ângulo entre vetores neste espaço (BERRY; DRMAC; JESSUP, 1999).

Segundo Cordeiro (2005), técnicas de análise semântica de textos procuram identificar a importância das palavras dentro da estrutura de suas orações. Porém, quando se utiliza um único texto algumas funções podem ser identificadas com um grau de importância. Entretanto, para algumas tarefas isso não é suficiente. Como exemplo, podem ser citadas as categorizações, onde é interessante analisar um documento comparando-o com bases de conhecimento de diferentes assuntos para descobrir a que categoria ele pertence.

A LSA consiste na construção de uma matriz A que informa a coocorrência de termos e documentos. Após, a matriz é algebricamente decomposta seguindo a SVD como forma de aproximação da matriz A por combinação linear. (BERRY; DRMAC; JESSUP, 1999).

Definindo formalmente, **A** é uma matriz onde o elemento *(i,j)* descreve a ocorrência de um termo *i* num documento *j*. Se **A** possui a dimensão *m x n*, *m* é o número total de termos avaliados e *n* é a quantidade de documentos avaliados, portanto a SVD de **A** é definida como:

$$\boldsymbol{A = USV^T}$$

Onde **U** = $[u_{ij}]$ é uma matriz ortonormal *mxm* das quais as colunas são chamadas de vetores singulares à esquerda; **S** = *diag*(σ1, σ2, ... σ*n*) é uma matriz diagonal de dimensão *mxn*, onde os elementos são chamados de valores singulares não negativos e que aparecem ordenados de forma crescente; e **V** = $[v_{ij}]$ é uma matriz ortonormal *nxn* onde as colunas são chamadas de vetores singulares à direita.

Se *posto*(**A**) = *k* a SVD pode ser interpretada como o mapeamento do espaço de **A** em um espaço conceito (reduzido) de k dimensões, as quais são linearmente independentes.

Ainda que *posto*(**A**) seja maior que k, quando seleciona-se apenas os k maiores valores singulares e os vetores singulares(σ1, σ2, ... σ*k*) e seus correspondentes vetores singulares em **U** e **V**, pode-se obter uma aproximação do posto *k* para a matriz **A** (GONG; LIU, 2001). Desta maneira, pode-se tratar os vetores de termos e os documentos como um espaço conceito. Neste espaço, os vetores de termos em **U** tem *k* entradas, cada um dando a ocorrência do termo *i* em um dos *k* conceitos.

Igualmente, os vetores de documentos em **V** revelam a relação entre o documento *j* com cada conceito *k*. Pode-se formalizar o espaço conceito como

$$\boldsymbol{A_k = U_k S_k V_k^T}$$

Deste modo, pode-se verificar em $\boldsymbol{V_k}$ o relacionamento entre dois dados documentos *j* e *l* (utilizando-se a distância de cosenos (BERRY; DRMAC; JESSUP, 1999)), possibilitando a aplicação de técnicas de agrupamento de documentos no espaço conceito.

A matriz de termos-documentos **A** carrega a informação sobre a ocorrência de termos em documentos. Geralmente emprega-se uma matriz de pesos sobre **A** de maneira a caracterizar a presença de um termo *i* em um documento *j*. Geralmente emprega-se uma matriz de pesos conhecida como TF-IDF (*term frequency - inverse document frequency*). Como parte da TF-IDF e com objetivo de medir a importância de um termo *i* em um documento *j*, emprega-se a frequência normalizada de um termo *i* em *j*:

$$TF_{i,j} = \frac{\#_{i,j}}{\sum_k \#_{k,j}}$$

Onde $\#_{i,j}$ é a quantidade de ocorrências do termo *i* no documento *j*, sendo o denominador a soma das ocorrências de todos os termos no documento *j*.

2.5 Softwares Relacionados

A MT já é utilizada na área de educação sob a forma de ferramentas cujos objetivos são auxiliar os alunos na produção textual. Cabral (2009) diz que os alunos possuem inúmeras dificuldades de expressão sob a forma textual, o que não acontece com a expressão oral, uma vez que a linguagem oral não exige formalidade na sua utilização, podendo ser usada a linguagem coloquial e gesticulações para facilitar o entendimento.

Pode-se citar alguns exemplos de ferramentas de mineração de texto que são utilizados na área da educação, como:

- SOBEK (SOBEK, 2015)
- TagCrowd (TAGCROWD, 2015)
- TextAlyser (TEXTALYSER, 2015)
- Word Counter (WORDCOUNTER, 2015)

O SOBEK é uma ferramenta capaz de extrair os termos mais frequentes em documentos, encontrando as relações entre eles, servindo de apoio para os professores na avaliação de trabalhos escritos (MACEDO *et al.*, 2009).

Ele se destaca pela apresentação gráfica dos resultados das relações entre os termos importantes nos documentos, auxiliando os alunos a formularem suas próprias compreensões sobre os textos lidos.

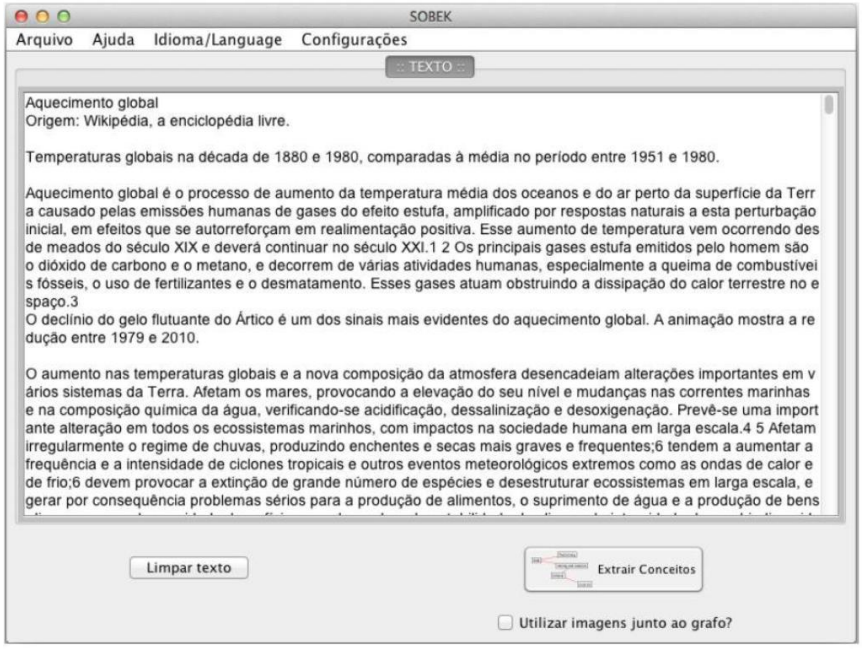


Figura 5: Tela de entradas de texto do SOBEK.

Fonte: http://sobek.ufrgs.br/uploads/2/3/3/9/23394804/sobek_quick_reference_guide_pt.pdf.

Na primeira tela do software pode ser digitado o texto a ser analisado, ou então carregar o texto a partir de documentos nos formatos *doc*, *pdf* ou *txt* (Figura 5). Após, ocorre a extração dos principais conceitos do texto, gerando uma representação em forma de grafo com os principais termos e relacionamentos entre eles (Figura 6).

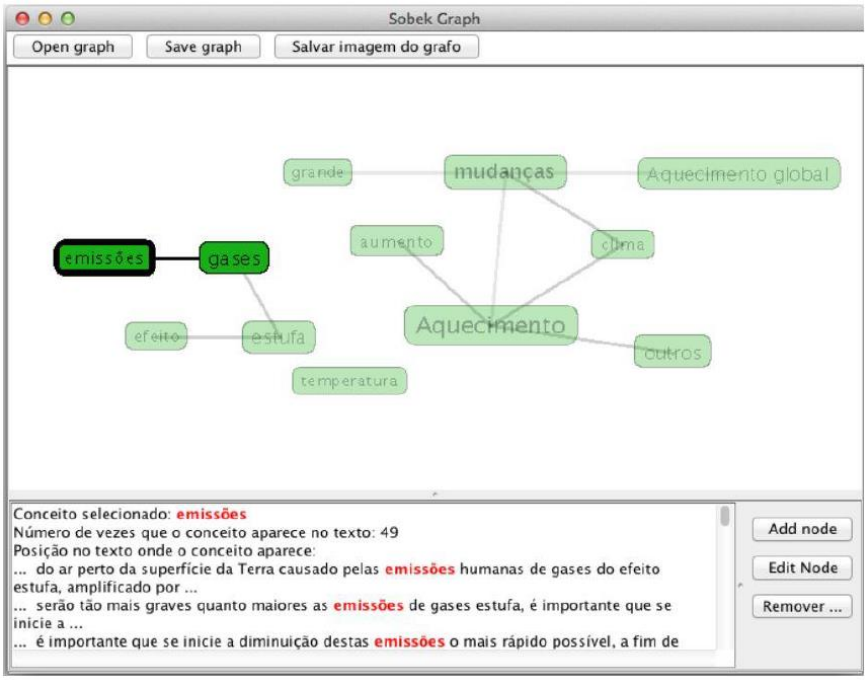


Figura 6: Tela de resultados do SOBEK.

Fonte: http://sobek.ufrgs.br/uploads/2/3/3/9/23394804/sobek_quick_reference_guide_pt.pdf.

O software TagCrowd (TAGCROWD, 2015) é uma ferramenta simples que permite criar uma representação gráfica no formato de nuvem de palavras, destacando palavras utilizadas com maior frequência nos textos analisados. No entanto, esta ferramenta não apresenta a relação entre os termos considerados importantes, e não remove os *StopWords* antes do processamento (Figura 7).

Figura 7: Tela de resultados do TagCrowd. Fonte: http://tagcrowd.com/.

O TextAlyser (TEXTALYSER, 2015) é uma ferramenta que analisa as palavras e expressões presentes no texto, apresentando a contagem e algumas estatísticas dos termos e palavras do documento analisado. Ela mostra ainda um índice de legibilidade do texto, avaliando critérios como tamanho das frases e estatísticas encontradas pela análise do texto.

O Word Counter é um software que também apresenta a relação das palavras mais utilizadas em um documento. Segundo Klemann, Reategui e Rapkiewicz (2011), esta ferramenta é de grande utilidade, pois apresenta as palavras repetidas ou redundantes em uma lista conforme Figura 8.

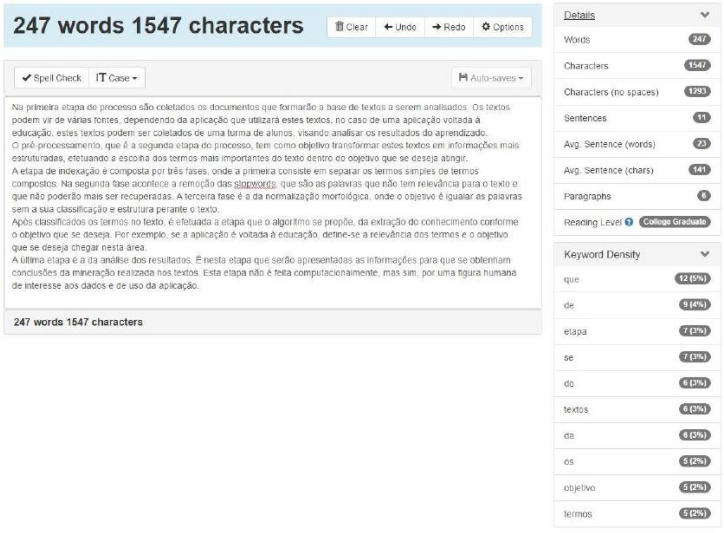


Figura 8: Tela do Word Counter Fonte: http://www.wordcounter.net/.

Klemann, Reategui e Rapkiewicz (2011) apresentam uma comparação entre estas ferramentas de MT, indicando claramente a intenção dos desenvolvedores de cada uma. Esta comparação pode ser visualizada na Tabela 1. Com relação à disponibilidade pela web, o SOBEK é a única ferramenta dentre as consideradas que não possui versão online. Somente os softwares TextAlyser e Word Counter possuem função de contagem de palavras. Quanto à apresentação dos termos analisados, os softwares TagCrowd e SOBEK apresentam apenas os termos considerados mais importantes, enquanto o Word Counter apresenta todos os termos. O software TagCrowd é o único que não possui a funcionalidade de identificar a frequência dos termos. O SOBEK é o único software capaz de analisar o relacionamento entre os termos. Referente à apresentação dos dados, os softwares SOBEK e TagCrowd possuem visualização gráfica dos termos, sendo que o SOBEK possui também a visualização gráfica dos termos relacionados. Os softwares TextAlyser e Word Counter não possuem nenhuma visualização gráfica.

Tabela 1: Comparação de ferramentas de MT conforme Klemann, Reategui e Rapkiewicz (2011)

	Online	Contagem de termos	Apresentação de todos os termos	Apresentação de termos relevantes	Frequência dos termos	Relacionamento dos termos	Vizualização gráfica dos termos	Visualização gráfica dos termos e relacionamentos
Sobek				X	X	X	X	X
TagCrowd	X			X			X	
TextAlyser	X	X			X			
Word Counter	X	X	X		X			

2.6 Considerações Finais

Conforme apresentado, a mineração de texto possui diversas aplicações, podendo ser utilizada nos mais variados domínios de conhecimento. Existem diferentes técnicas para analisar textos conforme os objetivos que se deseja atingir e informação requisitada. Foram analisadas algumas ferramentas existentes de MT e notou-se que a maioria delas não apresenta o relacionamento existente entre as palavras de um documento, e nenhuma faz a comparação entre documentos diferentes para utilização na avaliação de produções textuais no ambiente de educação. Esta é uma lacuna que este trabalho se propõe a suprir por meio da implementação do algoritmo LSA, cujo modelo é detalhado no capítulo seguinte.

3 Análise de Semântica Latente: Detalhamento e Exemplo de Funcionamento

Este capítulo detalha o algoritmo de latência semântica, exemplificando o seu funcionamento, permitindo visualizar os passos necessários para sua execução. Este estudo baseou-se em implementação disponível em linguagem Python, utilizando exemplos na língua inglesa (PUFFINWARRELC, 2012). O objetivo do autor é demonstrar a comparação de conceitos ligados às palavras mapeando elas com os documentos em um "espaço conceito", realizando as comparações neste espaço.

Segundo exemplo proposto pelo autor da implementação, considere livros pesquisados no site "*Amazon.com*" utilizando a palavra "*investing*". Considera-se neste exemplo os dez primeiros títulos encontrados. Um destes títulos foi desconsiderado por haver apenas uma única palavra índice em comum com os outros títulos. Uma palavra índice é qualquer palavra que aparece em dois ou mais títulos, e não é uma palavra *StopWord* (por exemplo, "*and*" e "*the*").

Neste exemplo, foram removidos *StopWords* "*and*", "*edition*", "*for*", "*in*", "*little*", "*of*", "*the*", "*to*". Os títulos restantes são os seguintes:

T1) The Neatest Little Guide to Stock Market Investing
T2) Investing For Dummies, 4th Edition
T3) The Little Book of Common Sense Investing: The Only Way to Guarantee Your Fair Share of Stock Market Returns
T4) The Little Book of Value Investing
T5) Value Investing: From Graham to Buffett and Beyond
T6) Rich Dad's Guide to Investing: What the Rich Invest in, That the Poor and the Middle Class Do Not!
T7) Investing in Real Estate, 5th Edition
T8) Stock Investing For Dummies
T9) Rich Dad's Advisors: The ABC's of Real Estate Investing: The Secrets of Finding Hidden Profits Most Investors Miss

Uma vez efetuada a LSA neste exemplo, pode-se plotar as palavras índices e os títulos num gráfico cartesiano com eixos X e Y e identificar os grupos de títulos. Os nove títulos estão plotados com círculos e as onze palavras índices como quadrados. É possível identificar os grupos de títulos e as palavras associadas aos grupos na Figura 9.

Por exemplo, o grupo que contém os títulos T7 e T9 estão relacionados por '*real estate'*. O grupo que contém os títulos T2, T4, T5 e T8 é sobre '*value investing'*, e o grupo que contém os títulos T1 e T3 é sobre '*stock market'*. O título T6 não está relacionado a nenhum outro título.

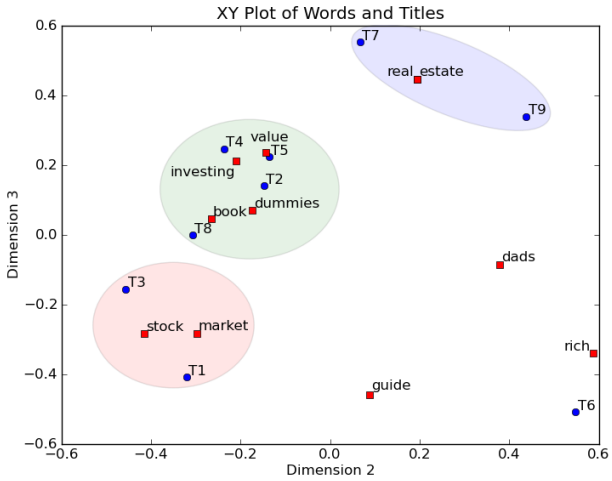


Figura 9: Gráfico de agrupamentos (PUFFINWARRELLC, 2012)

3.1 Criação da Matriz de Contagem

O primeiro passo do algoritmo LSA é criar a matriz *termo x documento*. Nesta matriz, cada palavra índice corresponde a uma linha e cada documento corresponde a uma coluna. Cada célula contém o número de ocorrências daquele termo no documento correspondente. Por exemplo, o termo "*book*" aparece uma vez no documento T3 e uma vez no documento T4, onde o termo "*investing*" aparece em todos os documentos. No geral, as matrizes geradas durante a execução do LSA tendem a ser grandes, porém esparsas. Isto se dá porque cada título ou documento contém um número reduzido de todas as palavras possíveis. Isto pode ser uma vantagem em implementações sofisticadas do algoritmo LSA. A matriz gerada pelo exemplo pode ser observada na Tabela 2, onde "*Titles*" corresponde aos documentos e "*Index Words*" corresponde aos termos analisados.

Tabela 2: Matriz Termo x Título

Index Words	**Titles**								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book	0	0	1	1	0	0	0	0	0
dads	0	0	0	0	0	1	0	0	1
dummies	0	1	0	0	0	0	0	1	0
estate	0	0	0	0	0	0	1	0	1
guide	1	0	0	0	0	1	0	0	0
investing	1	1	1	1	1	1	1	1	1
market	1	0	1	0	0	0	0	0	0
real	0	0	0	0	0	0	1	0	1
rich	0	0	0	0	0	2	0	0	1
stock	1	0	1	0	0	0	0	1	0
value	0	0	0	1	1	0	0	0	0

Primeiro, deve ser definida a informação utilizada. "*Titles*" contém os 9 títulos de livros considerados, *StopWords* contém as 8 palavras que serão desconsideradas dos títulos dos livros, e Ignorchars contém os caracteres de pontuação gramatical que serão removidos, como vírgula(,), ponto(.), apóstrofe(‘), e ponto de exclamação(!), exemplificado no código 1.

Código 1: definição de "*titles*", "*StopWords*" e "*ignorechars*"

```
titles =
        [
    "The Neatest Little Guide to Stock Market Investing",
    "Investing For Dummies, 4th Edition",
            "The Little Book of Common Sense Investing: The Only Way to Guarantee Your Fair Share of
            Stock Market Returns",
    "The Little Book of Value Investing",
    "Value Investing: From Graham to Buffett and Beyond",
            "Rich Dad's Guide to Investing: What the Rich Invest in, That the Poor and the Middle Class Do
            Not!",
    "Investing in Real Estate, 5th Edition",
    "Stock Investing For Dummies",
            "Rich Dad's Advisors: The ABC's of Real Estate Investing: The Secrets of Finding Hidden Profits
            Most Investors Miss"
    ]
stopwords = ['and','edition','for','in','little','of','the','to']
ignorechars = '",:'!'"
```

A classe LSA possui métodos para inicialização, percorrer documentos, montar a matriz de contagem de palavras, e cálculos. O primeiro método é o __init__, que é executado sempre que uma instância da classe LSA é criada. No código 2, verifica-se que são guardadas as *StopWords* e *ignorechars* para que possam ser utilizadas mais tarde, e então inicializar o dicionário de palavras e a contagem de variáveis de documento.

Código 2: *StopWords* e *ignorechars* são guardados em variáveis para serem acessadas posteriormente.

```
class LSA(object):
    def __init__(self, stopwords, ignorechars):
            self.stopwords = stopwords
        self.ignorechars = ignorechars
        self.wdict = {}
        self.dcount = 0
```

O método *parse* toma um documento, divide-o em palavras, retira os caracteres contidos na variável ignorechars e transforma todas as letras em minúsculas para que sejam comparadas com as *StopWords*. Caso não seja um *StopWord*, a palavra é inserida no dicionário e colocada junto com o número do documento para rastrear quais documentos contêm a palavra.

Os documentos em que cada palavra está contida são armazenados numa lista associada com a palavra no dicionário. Por exemplo, a palavra ‘*book’* aparece no título T3 e T4, portanto tem-se self.wdict[‘book’] = [3, 4] após percorrer todos os títulos.

Após processar todas as palavras do documento atual, incrementamos a contagem de documento para preparar para o próximo documento a ser percorrido. Este processo está descrito no código 3.

Código 3: Ocorre a separação do documento em seus termos componentes, sendo removidos *StopWords*. Termos novos são inseridos no dicionário de termos, e termos existentes têm seu contador incrementado.

```
def parse(self, doc):
        words = doc.split();
    for w in words:
            w = w.lower().translate(None, self.ignorechars)
          if w in self.stopwords:
                continue
           elif w in self.wdict:
                self.wdict[w].append(self.dcount)
           else:
                self.wdict[w] = [self.dcount]
        self.dcount += 1
```

Uma vez percorridos todos os documentos, todas as palavras que se encontram em mais do que um documento são extraídas e ordenadas, e uma matriz é construída com o número de linhas equivalentes ao número de palavras, e o número de colunas igual ao número de documentos. Por fim, cada tupla de palavra-documento tem sua célula correspondente incrementada. Este procedimento está descito no código 4.

Código 4: Construção da matriz Termo/Documento, com a frequência de cada termo em cada documento

```
def build(self):
    self.keys = [k for k in self.wdict.keys() if len(self.wdict[k]) > 1]
    self.keys.sort()
    self.A = zeros([len(self.keys), self.dcount])
    for i, k in enumerate(self.keys):
        for d in self.wdict[k]:
            self.A[i,d] += 1
```

3.2 Modificar Contagens com Frequência Inversa de Documentos

Na LSA, os valores da matriz são modificados para que palavras raras tenham maior peso que palavras comuns. Por exemplo, uma palavra que ocorre em apenas 5% dos documentos deveria ter peso maior do que uma palavra que ocorre em 90% dos documentos. A técnica mais comum para agregar esta função é Frequência Inversa de Documento (TFIDF). Com esta técnica, cada célula é substituída pela fórmula:

$TFIDFij = (N_{i,j} / N_{*,j}) * \log(D / D_i)$ onde

- Ni,j = número de vezes que uma palavra i aparece em um documento j
- N*j = número total de palavras em um documento
- D = número de documentos
- Di = número de documentos em que a palavra i aparece

Com esta fórmula, palavras concentradas em certos documentos são enfatizadas pela razão de $N_{i,j}$ / $N_{*,j}$ e palavras que aparecem em poucos documentos são enfatizadas pela razão de D / Di.

A utilização do TFIDF é representada no código 5. É necessário adicionar o seguinte método à classe LSA, onde *WordsPerDoc* (N*,j) contém a soma de cada coluna, que é o número total de palavras índice em cada documento; *DocsPerWord*(Di) utiliza um 'asarray' para criar um vetor de valores booleanos, armazenando se a célula possui valor maior que 0 ou não, armazenados como 1 para valor verdadeiro e 0 para valor falso. Após, cada linha é somada para indicar em quantos documentos cada palavra aparece. Por fim, basta percorrer as células aplicando a fórmula, transformando a variável 'cols' em *float* para evitar divisão de inteiros.

Código 5: Implementação do TFIDF

```
def TFIDF(self):
    WordsPerDoc = sum(self.A, axis=0)
    DocsPerWord = sum(asarray(self.A > 0, 'i'), axis=1)
    rows, cols = self.A.shape
    for i in range(rows):
        for j in range(cols):
            self.A[i,j] = (self.A[i,j] / WordsPerDoc[j]) * log(float(cols) / DocsPerWord[i])
```

Uma vez construída a matriz, utilize-se a técnica de SVD para analisar a matriz. Isto resultará numa representação dimensional reduzida da matriz com ênfase nas relações mais fortes, eliminando ruído. Ou seja, permite a melhor reconstrução da matriz com o mínimo de informação disponível. Para isto, valores não significativos são descartados da representação, permanecendo apenas os valores significativos.

O requisito para utilizar a SVD é a escolha do número de dimensões, ou conceitos, na aproximação da matriz. Se utilizar poucas dimensões, pode-se perder informação importante, no entanto, se utilizar muitas dimensões, não se elimina ruído causado por palavras aleatórias.

O Python possui uma biblioteca para executar o SVD. Adicionando as seguintes linhas de código 6 à classe LSA, pode-se fatorar a matriz em outras 3 matrizes. A matriz U contém as coordenadas de cada palavra no espaço-conceito. A matriz Vt contém as coordenadas de cada documento no espaço-conceito. A matriz S de valores singulares auxilia na escolha do número de dimensões necessárias.

Código 6: Utiliza-se da biblioteca SVD em Python para obter as matrizes U, S e Vt

```
def calc(self):
        self.U, self.S, self.Vt = svd(self.A)
```

Para escolher o número correto de dimensões a utilizar, pode-se criar um histograma do quadrado dos valores singulares. Isto demonstra a contribuição de cada valor singular na aproximação da matriz, como mostra a Figura 10.

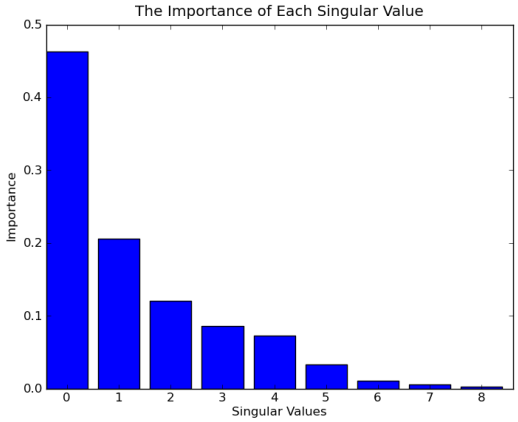


Figura 10: Histograma da importância de cada valor singular

Para números elevados de documentos, pode-se usar até 500 dimensões. Neste exemplo que será demonstrado graficamente, serão usados 3 dimensões, utilizando a segunda e terceira dimensão na representação gráfica.

A primeira dimensão está correlacionada ao comprimento total dos documentos quando referenciada por documentos, e à frequência de utilização quando referenciada por palavras. Se a matriz fosse centralizada subtraindo o valor médio de coluna de cada coluna, seria utilizada a primeira dimensão. Na LSA, não se centraliza a matriz porque transformaria uma matriz esparsa em uma matriz densa, aumentando o uso de memória e esforço computacional necessário. Portanto, é mais eficiente deixar a matriz descentralizada e desconsiderar a primeira dimensão.

A Figura 11 representa o resultado da decomposição SVD da matriz. Cada palavra possui três números associados, sendo um para cada dimensão. O primeiro valor tende a corresponder ao número de vezes que a palavra aparece em todos os títulos, portanto não é tão útil quanto as outras duas dimensões. De maneira similar, cada título possui três valores associados para cada dimensão. A primeira dimensão tende a corresponder ao número de palavras no título.

book	0.15	-0.27	0.04
dads	0.24	0.38	-0.09
dummies	0.13	-0.17	0.07
estate	0.18	0.19	0.45
guide	0.22	0.09	-0.46
investing	0.74	-0.21	0.21
market	0.18	-0.3	-0.28
real	0.18	0.19	0.45
rich	0.36	0.59	-0.34
stock	0.25	-0.42	-0.28
value	0.12	-0.14	0.23

*

3.91	0	0
0	2.61	0
0	0	2

*

T1	T2	T3	T4	T5	T6	T7	T8	T9
0.35	0.22	0.34	0.26	0.22	0.49	0.28	0.29	0.44
-0.32	-0.15	-0.46	-0.24	-0.14	0.55	0.07	-0.31	0.44
-0.41	0.14	-0.16	0.25	0.22	-0.51	0.55	0	0.34

Figura 11: Decomposição SVD da matriz resulta na matriz U, S e Vt

Após a compreensão do algoritmo LSA pode-se partir para a sua implementação no contexto Under Mine Text Miner. O Capítulo 4 a seguir apresenta o desenvolvimento e a integração do algoritmo LSA à plataforma existente.

4. Ferramenta de Análise de Produções Textuais

Este capítulo tem por objetivo apresentar a ferramenta Análise de Produções Textuais. Ela é uma ferramenta capaz de efetuar o treinamento sobre determinadas áreas de conhecimento, fazendo então a análise de textos produzidos por alunos e apresentar os resultados obtidos. Foi desenvolvida uma nova ferramenta de análise dos dados visando estender a funcionalidade de mineração de textos presentes no projeto precedente.

4.1 Modelagem

No trabalho de conclusão desenvolvido por Lovato (2015), o objetivo era a visualização dos resultados obtidos pelo treinamento do algoritmo sobre determinada área de estudo e a subsequente análise e classificação dos artigos. Este sistema possuía duas etapas que eram executadas separadamente. A primeira era a etapa de treinamento, onde ocorre a aprendizagem dos *tokens* dos artigos utilizados. A segunda etapa é de Análise, onde ocorre a análise dos textos dos alunos e então são apresentados os resultados visuais. Estas duas etapas são apresentadas na mesma aplicação, separadas por abas.

Seguindo a lógica utilizada por Lovato (2015), foi mantida a estrutura básica da aplicação, sendo necessário adicionar uma seleção de valor K para determinar a precisão dos resultados do algoritmo de LSA, e a adição de uma coluna que representa a nota resultante da análise LSA. Foi também adicionada uma nova modalidade de visualização de dados em formato *ScatterPlot*.

O sistema Análise de Produções Textuais implementa as mesmas camadas que o trabalho desenvolvido por Lovato (2015), sendo necessário apenas adicionar as classes LeitorLSA e LSAMain, responsáveis pela preparação dos dicionários e tabelas utilizadas pelo algoritmo, e a execução do algoritmo LSA, respectivamente. Foram necessárias algumas alterações nas camadas existentes que serão descritas no capítulo 4.3. A Tabela 3 apresenta as classes do sistema.

Tabela 3: Classes do sistema Análise de Produções Textuais, Adaptados de Lovato (2015)

Nome da Classe	Objetivos
Treinamento	
Mensagem	- Representa os atributos do objeto do tipo Mensagem.
Token	- Representa os atributos do objeto do tipo Token.
Arquivo	- Classe responsável pela leitura dos textos para treinamento do sistema.
Artigo	- Classe centralizadora das lógicas do treinamento.
StopWord	- Classe responsável por remover as *StopWords* dos textos.

Análise	
AgenteDecompositor	- Decompor o artigo em *tokens*. - Gerar Listas com os termos decompostos
VerificadorConcentracaoTokens	Calcular a concentração dos *tokens* no artigo.
AgenteVerificadorDiagnostico	- Utilizando os dados das verificações anteriores, realiza diagnóstico do artigo.
AnaliseProdText	- Classe de interface do programa principal.
AlteraConfig	- Classe de interface para configuração do sistema.
Diagnosticar	- Classe que carrega dados necessários para o sistema e inicializa os agentes.
QuadroNegro	- Classe onde todos os agentes escrevem informações. Essa classe é acessível para todo o sistema.
Artigo	- Classe que representa os atributos de uma produção textual de um aluno.
ListaArtigo	- Classe que armazena as produções textuais dos alunos a serem analisadas após terem sido lidas pelo sistema.
CarregarDados	- Classe responsável pela leitura das produções textuais dos alunos.
Termos	- Representa os atributos do objeto do tipo Termo.
Token	- Representa os atributos de um objeto do tipo Token.
ResultadoDiagnóstico	- Representa os atributos de um objeto do tipo ResultadoDiagnóstico. Nesta classe ficam armazenados os resultados da análise em tempo de execução.
LeitorLSA	-Classe que prepara dicionários e matrizes para executar algoritmo LSA
LSAMain	- Classe que gera as matrizes SVD e matrizes derivadas - Executa algoritmo LSA - Executa comparação de textos com material de treinamento
Visualizações	
BarChart	- Classe responsável por estruturar os dados necessários para a visualização do tipo Bar Chart e efetuar a chamada da respectiva visualização.
BubbleChart	- Classe responsável por estruturar os dados necessários para a visualização do tipo Bubble Chart e efetuar a chamada da respectiva visualização.
CollapsibleTree	- Classe responsável por estruturar os dados necessários para a visualização do tipo Collapsible Tree ou Rotating Cluster e efetuar a chamada da respectiva visualização.
TreeMap	- Classe responsável por estruturar os dados necessários para a visualização do tipo TreeMap e efetuar a chamada da respectiva visualização.
WordCloud	- Classe responsável por estruturar os dados necessários para a visualização do tipo Word Cloud e efetuar a chamada da respectiva visualização.

No anexo A é apresentado um diagrama de classes do sistema, facilitando a visualização detalhada dos atributos de cada classe.

4.2 Arquitetura

Tendo em vista que a ferramenta está inserida em um projeto preexistente, é interessante manter as plataformas utilizadas anteriormente para garantir que as funcionalidades originais sejam compatíveis. Assim, a ferramenta foi desenvolvida para *desktop*, com as visualizações *web* em JavaScript. Existindo essa relação entre *desktop* e *web*, é necessário utilizar um servidor *web*.

Ao ser inicializado, o sistema funciona totalmente como *desktop*, tanto a funcionalidade de treinamento do software quanto a análise das produções textuais. Após a análise dos textos e execução do algoritmo LSA no *desktop*, desenvolvida em C#, as visualizações são disponibilizadas no *browser* padrão do usuário, visto que foram desenvolvidos com base na tecnologia *D3 Data-Driven Documents* (BOSTOCK, 2013) que se vale da linguagem JavaScript para gerar imagens SVG, e HTML e CSS para apresentá-las.

A comunicação entre a camada *desktop* e as visualizações *web* é feita através de arquivos no formato *JSON* e *TSV*, que são salvos no diretório de publicação do servidor *web*. Deste modo, podem ser lidos pelas aplicações desenvolvidas para a camada *web*.

4.3 Implementação

A descrição da implementação contempla as alterações necessárias em relação à ferramenta de Lovato (2015). Também descreve o algoritmo de LSA que foi implementado e integrado à ferramenta preexistente.

O sistema oferece uma análise diferenciada das produções textuais. O algoritmo LSA foi implementado em 11 etapas, sendo parte delas executada no treinamento e o restante ocorrendo no diagnóstico. Tornou-se necessário efetuar a etapa de treinamento para que os dados necessários para a execução da LSA sejam buscados e preparados.

4.3.1 Treinamento

A etapa de treinamento original foi modificada para permitir a captação e preparação dos textos de treinamento para que pudessem ser armazenados em formato de *HashTable*. Uma *HashTable* é uma estrutura de dados que permite associar um objeto a um índice numérico. Este procedimento mapeia cada documento analisado a um identificador numérico,

necessário para poder identificar cada documento individual na matriz resultante do algoritmo LSA. Foi incluído um método na classe *StopWord* para verificar se uma palavra pertence à lista de *StopWords*, que se encontram na Tabela 4.

Após efetuar a leitura de cada documento, cria-se uma lista geral de cada termo utilizado, sendo filtrados os *StopWords*. Cada termo lido é mapeado numa *HashTable* com identificador numérico próprio, e utilizado para criar uma relação de Termo/Documento, onde se relacionam os termos utilizados em cada documento, assim como a frequência do termo.

O programa apresenta uma interface gráfica ilustrada na Figura 12. Primeiro informa-se a atividade, o diretório que contém os textos utilizados no treinamento do software e o idioma. Ao clicar no botão "Processar Artigos" ocorre a execução do processo.

Além dessas mudanças, foram efetuados alguns aprimoramentos a fim de permitir ajustes no tamanho da tela, sem perder a formatação das informações presentes nela. Estes ajustes foram os pontos de ancoragem dos itens presentes na tela, e os caracteres de formatação das informações escritas.

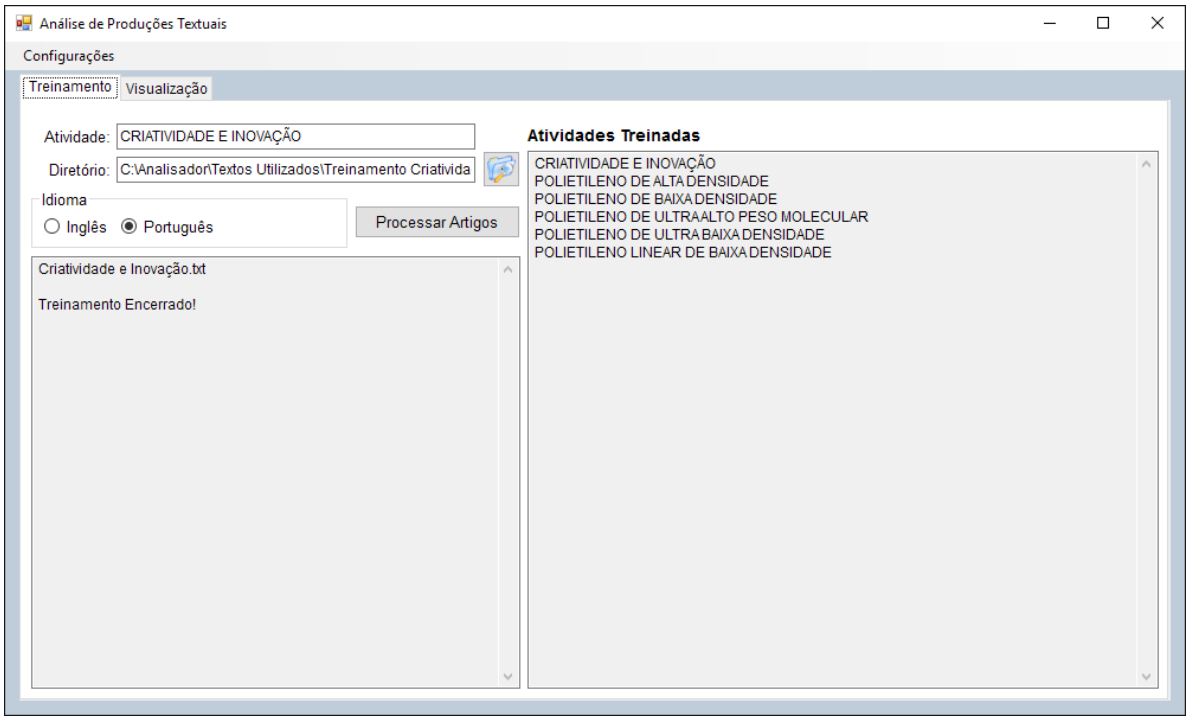


Figura 12: Treinamento da ferramenta Análise de Produções Textuais.

Tabela 4: Tabela de *StopWords*

Lista de Stop Words					
A	DESSA	ESTAMOS	MUITAS	PODER	SOBRE
À	DESSAS	ESTÃO	MUITO	PODERIA	SUA
AGORA	DESSE	ESTAS	MUITOS	PODERIAM	SUAS
AINDA	DESSES	ESTAVA	NA	PODIA	TALVEZ
ALGUÉM	DESTA	ESTAVAM	NÃO	PODIAM	TAMBÉM
ALGUM	DESTAS	ESTÁVAMOS	NAS	POIS	TAMPOUCO
ALGUMA	DESTE	ESTE	NEM	POR	TE
ALGUMAS	DESTE	ESTES	NENHUM	PORÉM	TEM
ALGUNS	DESTES	ESTOU	NESSA	PORQUE	TENDO
AMPLA	DEVE	EU	NESSAS	POSSO	TENHA
AMPLAS	DEVEM	FAZENDO	NESTA	POUCA	TER
AMPLO	DEVENDO	FAZER	NESTAS	POUCAS	TEU
AMPLOS	DEVER	FEITA	NINGUÉM	POUCO	TEUS
ANTE	DEVERÁ	FEITAS	NO	POUCOS	TI
ANTES	DEVERÃO	FEITO	NOS	PRIMEIRO	TIDO
AO	DEVERIA	FEITOS	NÓS	PRIMEIROS	TINHA
AOS	DEVERIAM	FOI	NOSSA	PRÓPRIA	TINHAM
APÓS	DEVIA	FOR	NOSSAS	PRÓPRIAS	TODA
AQUELA	DEVIAM	FORAM	NOSSO	PRÓPRIO	TODAS
AQUELAS	DISSE	FOSSE	NOSSOS	PRÓPRIOS	TODAVIA
AQUELE	DISSO	FOSSEM	NUM	PUDE	TODO
AQUELES	DISTO	GRANDE	NUMA	QUAIS	TODOS
AQUILO	DITO	GRANDES	NUNCA	QUAL	TU
AS	DIZ	HÁ	O	QUANDO	TUA
ATÉ	DIZEM	ISSO	OS	QUANTO	TUAS
ATRAVÉS	DO	ISTO	OU	QUANTOS	TUDO
CADA	DOS	JÁ	OUTRA	QUE	ÚLTIMA
COISA	E	LA	OUTRAS	QUEM	ÚLTIMAS
COISAS	É	LÁ	OUTRO	SÃO	ÚLTIMO
COM	ELA	LHE	OUTROS	SE	ÚLTIMOS
COMO	ELAS	LHES	PARA	SEJA	UM
CONTRA	ELE	LO	PELA	SEJAM	UMA
CONTUDO	ELES	MAS	PELAS	SEM	UMAS
DA	EM	ME	PELO	SEMPRE	UNS
DAQUELE	ENQUANTO	MESMA	PELOS	SENDO	VENDO
DAQUELES	ENTRE	MESMAS	PEQUENA	SERÁ	VER
DAS	ERA	MESMO	PEQUENAS	SERÃO	VEZ
DE	ESSA	MESMOS	PEQUENO	SEU	VINDO
DELA	ESSAS	MEU	PEQUENOS	SEUS	VIR
DELAS	ESSE	MEUS	PER	SI	VOS
DELE	ESSES	MINHA	PERANTE	SIDO	VÓS
DELES	ESTA	MINHAS	PODE	SÓ	
DEPOIS	ESTÁ	MUITA	PODENDO	SOB	

4.3.2 Análise de Produções Textuais

A etapa de análise das produções textuais sofreu alterações perceptíveis para se adequar aos objetivos propostos. Visto que o objetivo é estender a funcionalidade do software para incluir a análise LSA, a tela responsável pela apresentação dos resultados sofreu algumas alterações para permitir realizar a análise e visualizar o resultado.

Conforme pode ser visto na Figura 13, foram incluídos alguns itens necessários para o correto funcionamento da análise LSA, como o "*spinbox*" para a escolha do "valor K", que será explicado no capítulo 4.4, a coluna "NotaLSA" para demonstrar resumidamente o resultado da análise, e o botão "*LSA ScatterPlot*" que gera uma visualização mais completa do resultado. A ferramenta cria um arquivo TSV com os resultados obtidos e armazena na pasta correspondente ao *ScatterPlot* no *webserver*. O D3 por sua vez, utiliza o arquivo como fonte de dados ao abrir o arquivo *HTML* correspondente à visualização *ScatterPlot* onde está contido o código JavaScript que gera as visualizações.

Nesta tela foram efetuados os mesmos aprimoramentos visuais com o propósito de permitir alterar o tamanho da tela sem desestruturar as informações nela contidas. Os pontos de ancoragem foram redefinidos para que fosse possível aumentar a área de visualização dos resultados finais.

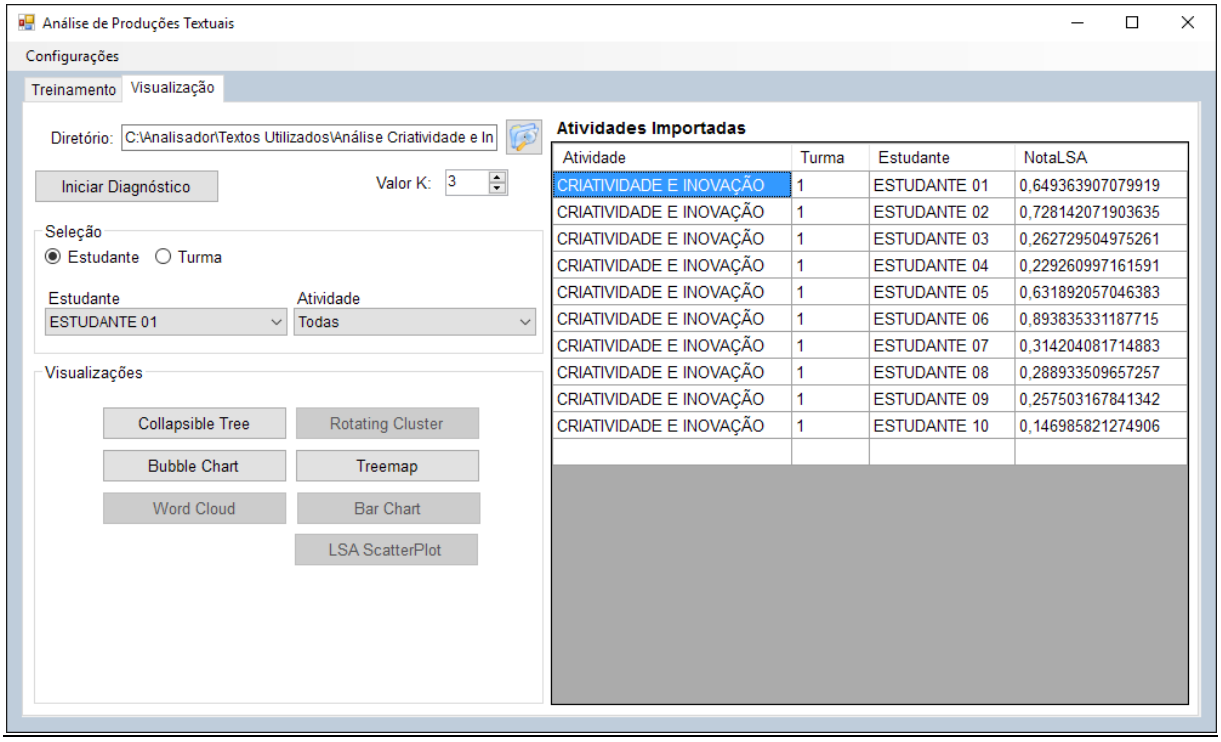


Figura 13: Tela de análise da ferramenta Análise de Produções Textuais

4.4 Algoritmo LSA

O funcionamento do algoritmo LSA implementado se dá em 11 etapas que são descritas a seguir. No desenvolvimento deste algoritmo, foi utilizada a biblioteca "*DotNetMatrix*" para a manipulação das matrizes geradas pelo software.

A primeira etapa consiste na inicialização das *HashTables* que são utilizadas no decorrer do processo de LSA. As *HashTables* são: "*goWordList*", "*goDocList*", e "*goDocWord*", onde são indexados termos, documentos e a relação termo/documento, respectivamente. Nesta etapa ocorre também a leitura do texto de treinamento, contando a frequência de cada termo.

Na segunda etapa é iniciada a indexação do documento de treinamento. Neste processo os *StopWords* são filtrados, os termos restantes são inseridos na *HashTable* "goWordList". Utilizando os índices de cada termo é gerada a relação Termo/Documento junto à frequência de cada termo. Por fim, insere-se o documento de treinamento nas *HashTable goDocList* no índice 0.

A terceira etapa é similar à segunda, porém indexa-se os textos a serem analisados. O processo é o mesmo: filtragem dos *StopWords*, inserção de novos termos na *HashTable* "goWordList", e geração das relações Termo/Documento e suas respectivas frequências. Por fim, cada documento é inserido na *HashTable* "goDocList" em um novo índice.

O quarto passo do algoritmo é a criação da matriz Termo/Documento, reunindo todos os termos utilizados com todos os documentos indexados. Cada linha desta matriz corresponde a um termo, e cada coluna corresponde a um documento. Cada célula da matriz corresponde à frequência presente nas relações Termo/Documento, geradas nas etapas anteriores. Estas frequências são conhecidas como pesos locais.

Na quinta etapa ocorre o cálculo do peso global de cada termo. O peso global é uma função de quantos documentos dentro da coleção contêm a palavra. A função utilizada foi a frequência inversa de documentos.

A sexta etapa consiste em calcular os fatores de normalização dos termos de cada documento. Para cada documento, soma-se o quadrado da multiplicação entre o peso global do termo e o peso local do termo no respectivo documento de todos os termos. O fator de normalização é obtido dividindo 1 pela raiz quadrada da soma.

Na sétima etapa é criada a matriz Termo/Documento Ponderada, ou seja, é a matriz que contém todos os termos e todos os documentos, porém o valor de cada célula possui o valor modificado para contemplar o peso local de cada termo, o peso global de cada termo, e o fator de normalização de cada documento.

Na oitava etapa do algoritmo LSA ocorre o cálculo da Decomposição em Valores Singulares (SVD). Este cálculo resulta nas matrizes S, U e V, sendo que a partir da

multiplicação destas matrizes é possível reconstruir a matriz Termo/Documento Ponderada original, seguindo A = U *x* S *x* Vt.

O nono passo do algoritmo LSA consiste na preparação das matrizes reduzidas, utilizadas no cálculo de similaridade dos documentos. Estas matrizes são Sk e Uk. As matrizes reduzidas são calculadas baseadas no valor K definido na interface gráfica. As matrizes Sk e Uk nada mais são que as primeiras K linhas e colunas das matrizes S e U respectivamente.

A escolha do valor K é importante, uma vez que é utilizado para reduzir a dimensionalidade da matriz Termo/Documento ponderada. No entanto, a escolha de um valor K muito pequeno, como 2, embora deixe o desempenho do cálculo melhor, resulta num cálculo muito impreciso devido à falta de informação. Em contrapartida, a escolha de um valor K alto demais, como a contagem de documentos analisados, introduz muito ruído no cálculo, prejudicando também a precisão do resultado final. A escolha do valor K é, portanto, muito importante para obter resultados viáveis. Um valor inicial recomendado é de 30% da quantidade de documentos a ser analisados.

Na décima etapa, ocorre o armazenamento em disco de diversas variáveis necessárias para o cálculo de similaridade de documentos no próximo passo do algoritmo. Este passo existe para diminuir o uso de memória durante o processamento, uma vez que o conjunto de matrizes pode crescer rapidamente.

Na última etapa é criada uma matriz contendo apenas o vetor de termos correspondente ao documento de treinamento, obtidos da matriz Termo/Documento Ponderada. Esta matriz então é transposta e multiplicada pela matriz Uk e novamente multiplicada pela matriz Skinversa. Este procedimento resulta numa representação reduzida do documento de treinamento. O mesmo processo é executado para cada documento analisado, sendo que cada representação reduzida é utilizada para calcular a similaridade entre o documento analisado e o documento de treinamento.

O cálculo de similaridade entre os documentos pode ser definido como sendo a comparação entre os vetores gerados a partir das matrizes reduzidas, onde a proximidade dos vetores é proporcional à similaridade entre os documentos. Para obter o resultado numérico, é calculado o cosseno do ângulo entre cada vetor, armazenado em " NotaLSA". Na Figura 14 é demonstrada a similaridade entre vetores *q*, *d1*, *d2* e *d3*.

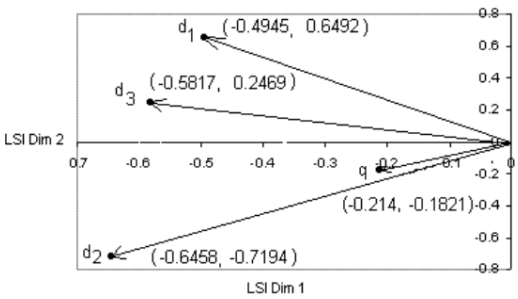


Figura 14: Exemplo de mapeamento de vetores LSA

4.5 Testes

A sessão de testes visa apresentar o funcionamento da ferramenta Análise de Produções Textuais e os testes realizados. O objetivo é apresentar as etapas de utilização da funcionalidade de análise LSA.

Para este cenário de teste foram utilizados os mesmos textos de treinamento da ferramenta, utilizados por Lovato (2015), que incluem: Criatividade e Inovação, Polietileno de baixa densidade (PEBD ou LDPE), Polietileno de alta densidade (PEAD ou HDPE), Polietileno linear de baixa densidade (PELBD ou LLDPE), Polietileno de ultra alto peso molecular (PEUAPM ou UHMWPE) e Polietileno de ultra baixa densidade (PEUBD ou ULDPE). A Figura 15 demonstra um arquivo utilizado para treinar a ferramenta, neste caso apresentando Criatividade e Inovação.

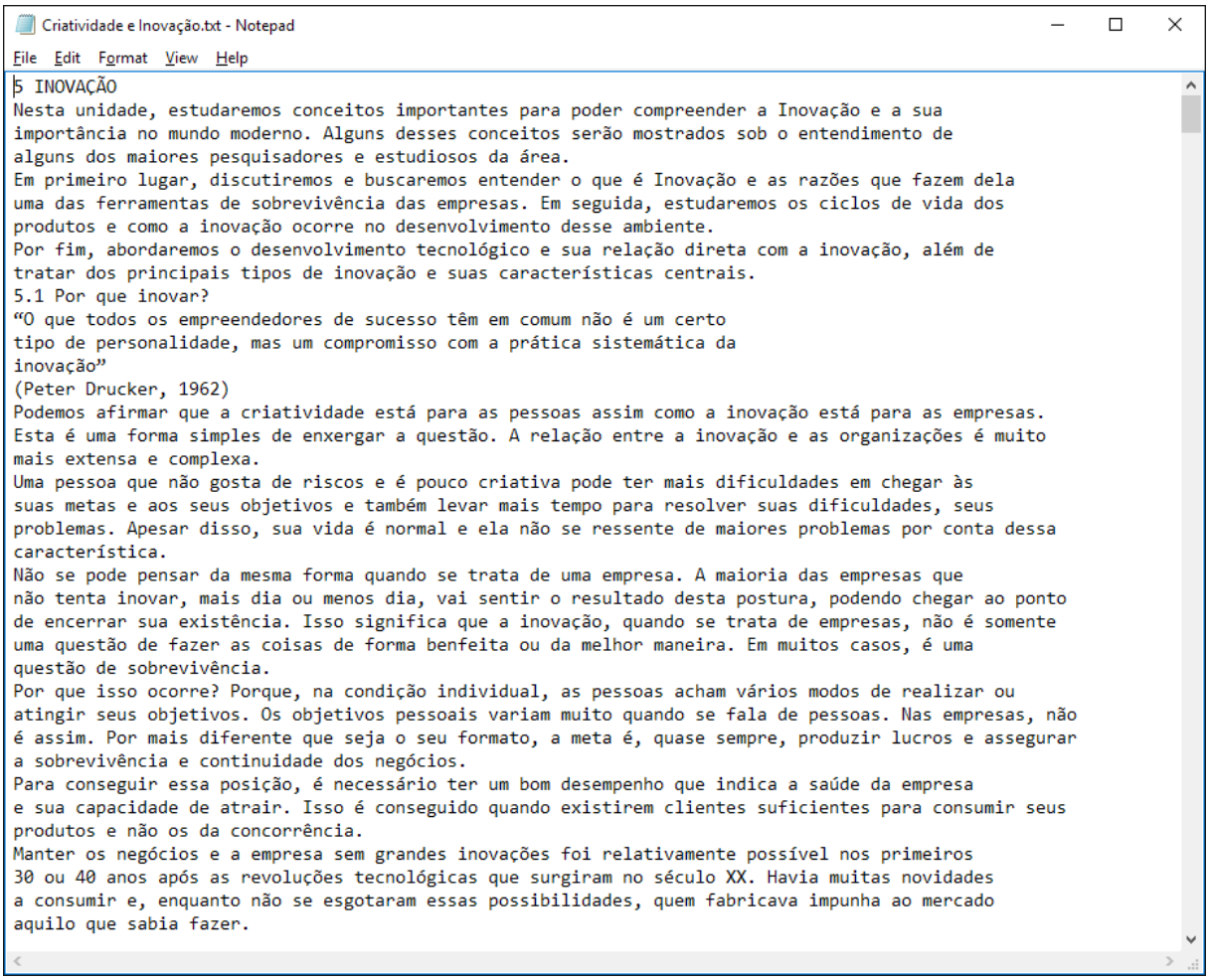


Figura 15: Texto utilizado para o treinamento de Criatividade e Inovação

Após o treinamento da ferramenta, são submetidas as produções textuais dos alunos. A Figura 16 apresenta um exemplo de produção textual utilizado a ser submetido à análise da ferramenta. O cabeçalho utilizado para identificar a atividade, a turma e o aluno permanecem iguais aos utilizados por Lovato (2015), sendo necessário para que a ferramenta possa classificar os textos e criar as visualizações correspondentes.

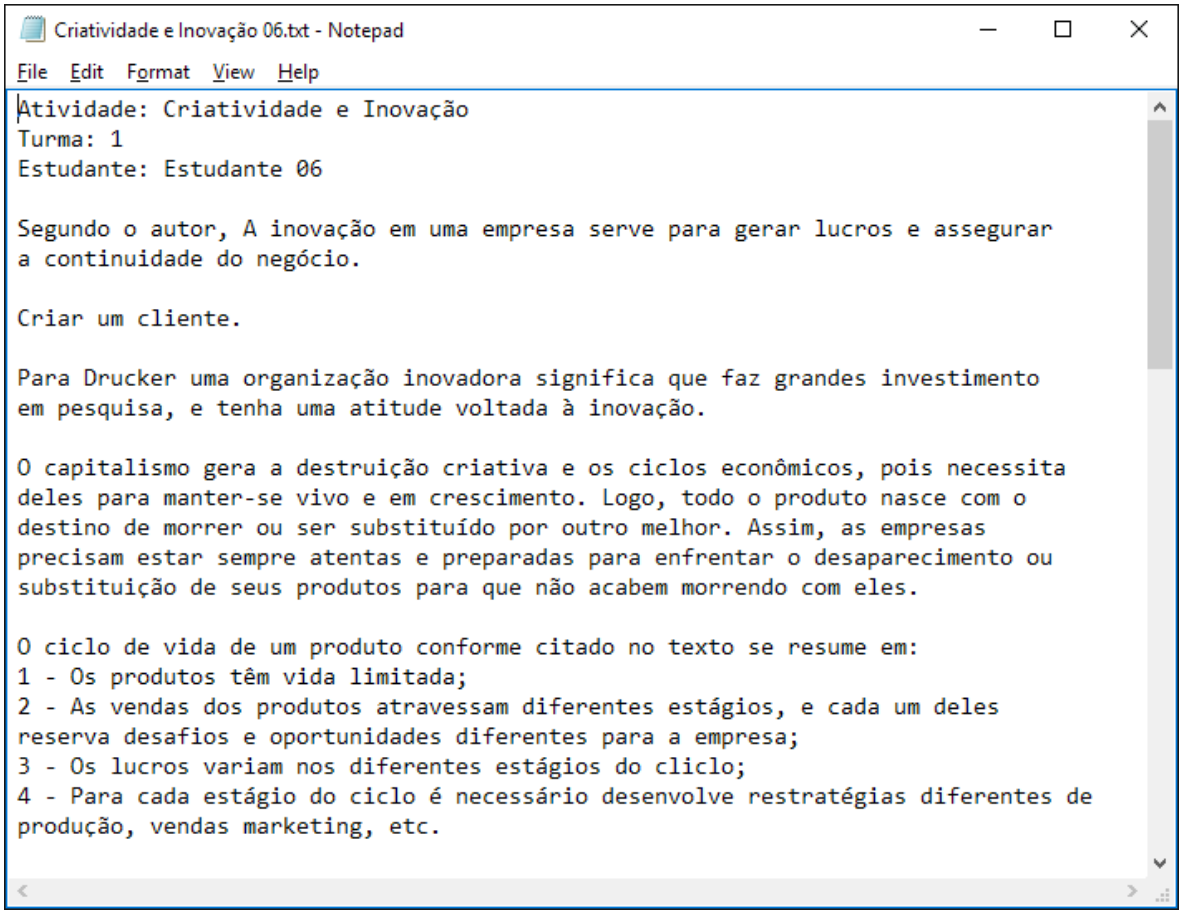


Figura 16: Exemplo de produção textual a ser analisado

4.5.1 Testes de Funcionamento

Antes de utilizar a ferramenta para seu proposto objetivo, é necessário avaliar alguns casos de uso para verificar o correto funcionamento da mesma. Como teste, foram utilizados valores de K = 1 e K = 100. Estes valores foram escolhidos por saber-se que fornecerão resultados incorretos, uma vez que é impossível calcular o ângulo entre vetores de dimensão 1, e por ser impossível gerar uma matriz LSA com valor K maior que a quantidade de

documentos a analisar. Foi também utilizado o texto de treinamento como produção textual para verificar que a avaliação de similaridade está funcionando corretamente.

Como pode ser visto na Figura 17, a utilização de valor K pequeno demais provoca uma perda de precisão, inutilizando os resultados gerados pelo algoritmo LSA. Isto é causado pela redução da matriz Termo/Documento ponderada a uma única dimensão, portanto ao comparar a diferença de ângulo entre 2 pontos, o resultado é 0. Por isso, foi determinado que o valor mínimo de K para se obter resultados utilizáveis é 2.

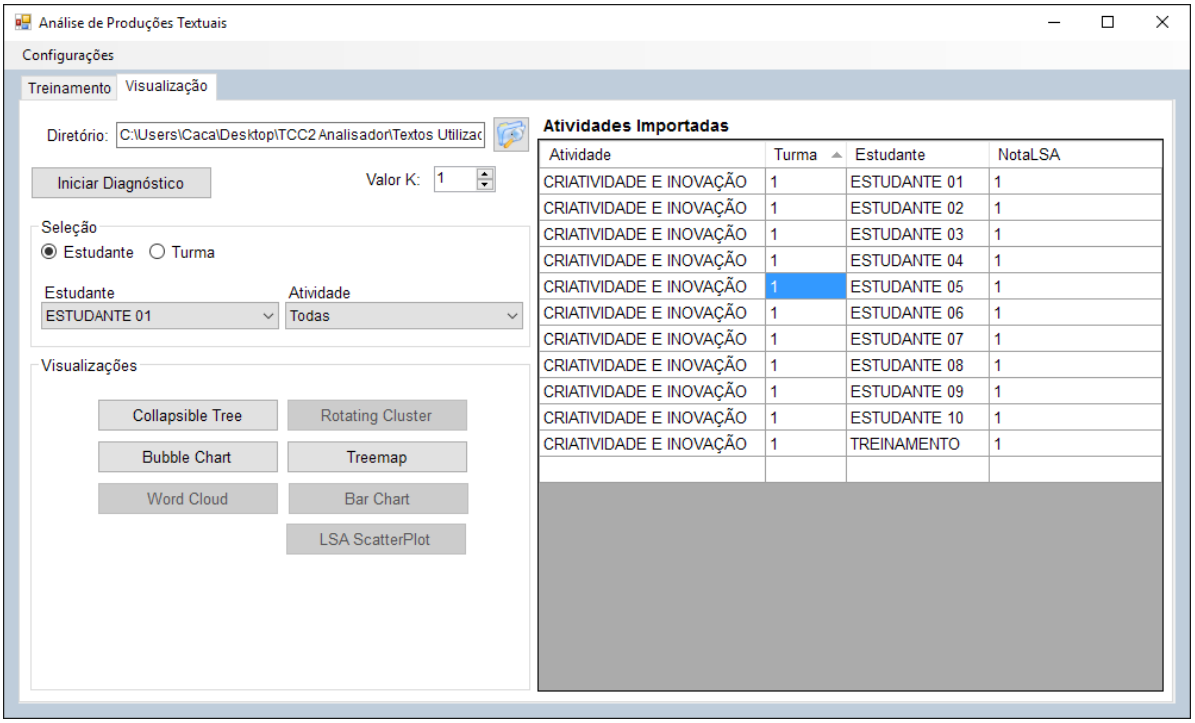


Figura 17: Demonstração utilizando Valor K = 1

Já na Figura 18 pode-se ver o efeito da escolha de um valor K grande demais. Neste caso, o valor K escolhido é maior que o número de documentos analisados, portanto é utilizada a contagem de documentos como valor K. No entanto, o ruído introduzido no cálculo faz com que o resultado obtido não seja utilizável, uma vez que os resultados estão todos próximos de 0.

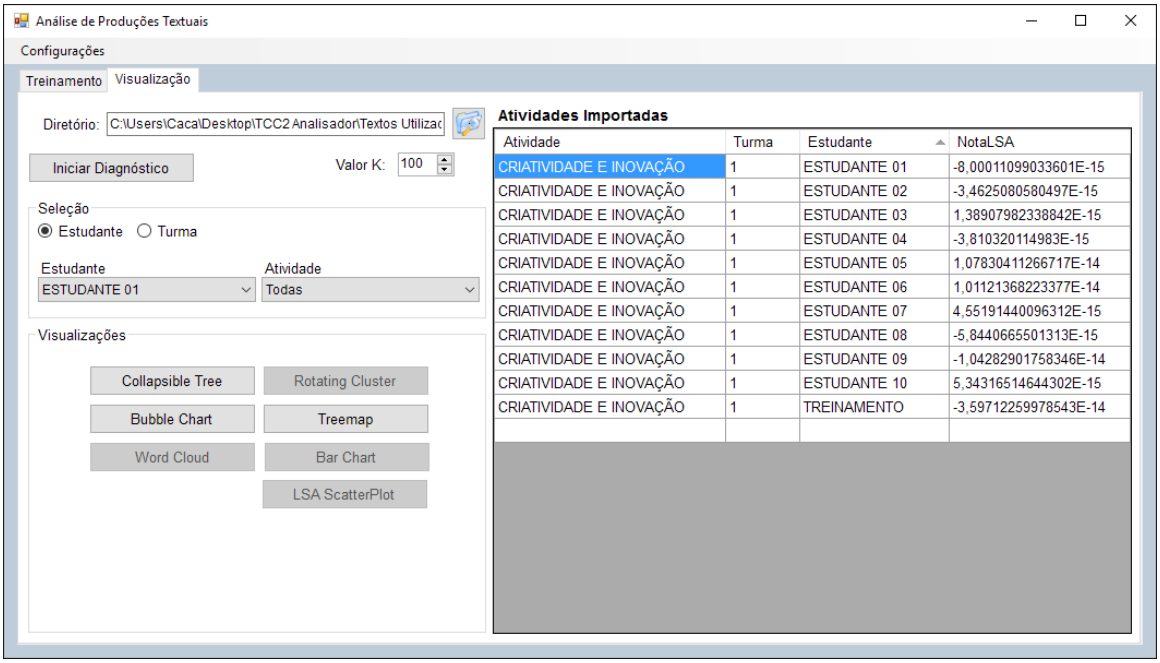


Figura 18: Demonstração com Valor K = 100

A Figura 19 demonstra, na coluna "NotaLSA", a nota fornecida pela ferramenta ao analisar o próprio texto de treinamento como produção textual. Como o resultado é muito próximo ao valor de 1.0, conclui-se que o resultado gerado pela ferramenta está de acordo com o esperado. A pequena diferença pode ser explicada pela presença do cabeçalho no arquivo analisado.

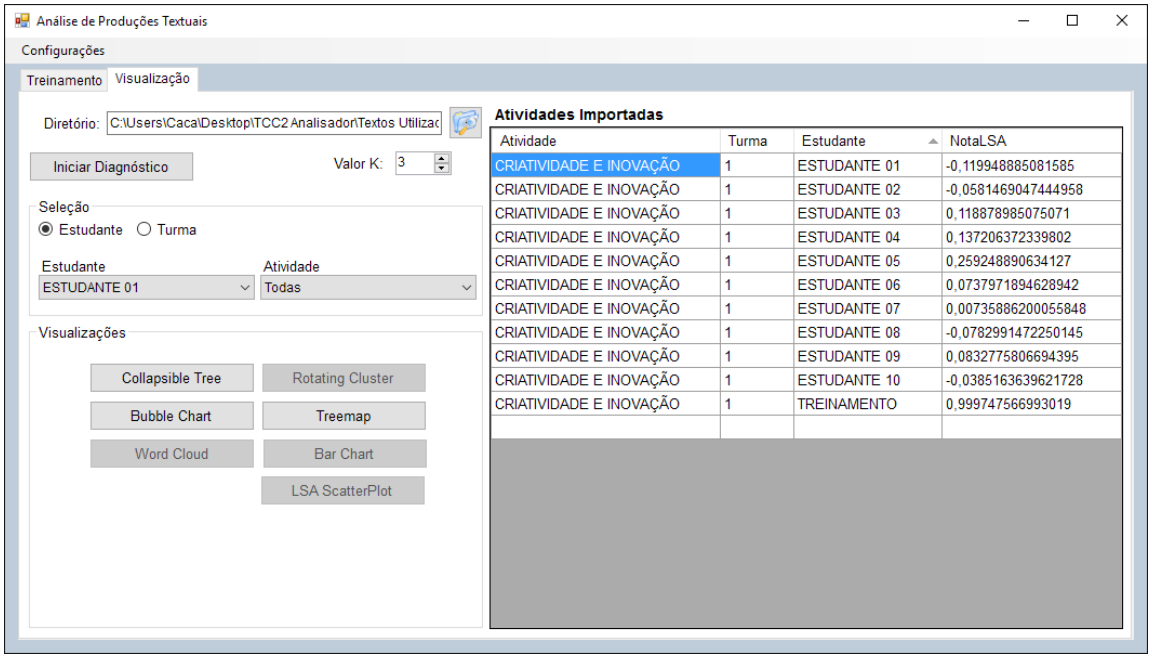


Figura 19: Demonstração de resultado ao analisar documento de treinamento

4.5.2 Testes da Turma de Sistemas Informação e Decisão

Para avaliação da ferramenta em desenvolvimento foram utilizados textos de uma turma da disciplina de Sistemas de Informação e Decisão. Esta turma teve como atividade a leitura do assunto Criatividade e Inovação, seguido pela resolução de um questionário. O sistema Análise de Produções Textuais foi treinado sobre o assunto e o questionário dos alunos foi submetido à análise.

Como pode-se observar na Figura 20, foi constatado que o ESTUDANTE 06 atingiu o maior resultado da turma, enquanto o ESTUDANTE 10 atingiu o menor resultado da turma. Assim, de acordo com estes valores, o ESTUDANTE 06 produziu um texto mais semelhante ao texto utilizado no treinamento da turma. Comparando a produção textual do ESTUDANTE 06 na Figura 21 à produção textual do ESTUDANTE 10 na Figura 22, fica evidente a diferença na qualidade da escrita de cada estudante.

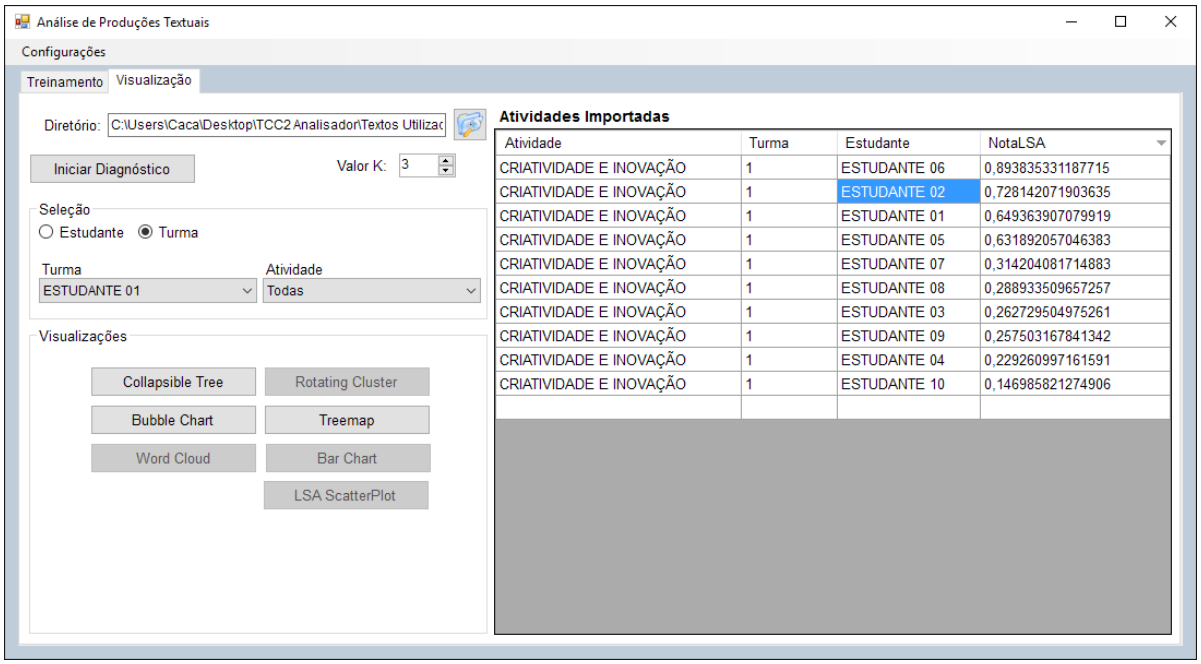
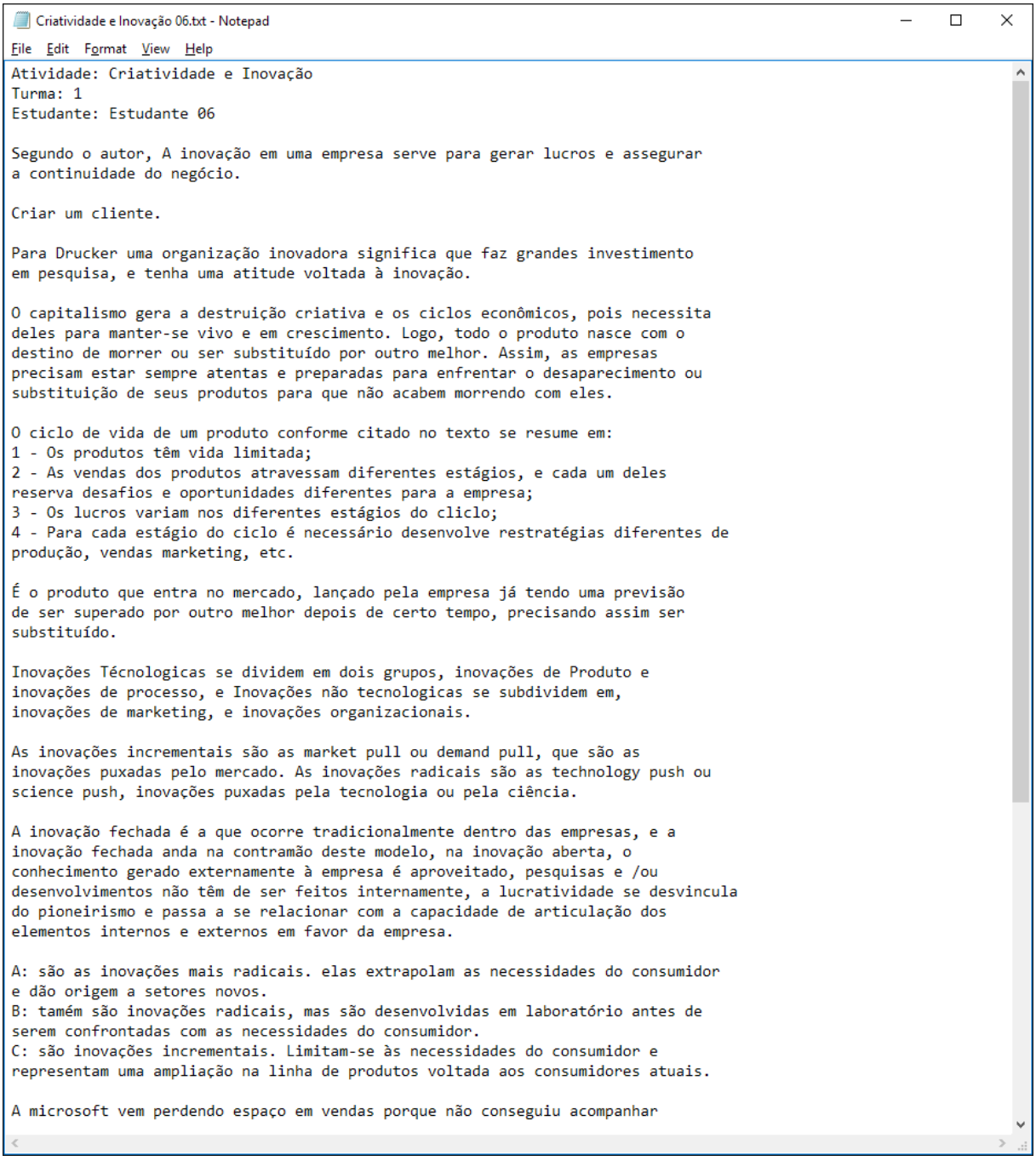


Figura 20: Análise da Turma de Informação e Decisão



Criatividade e Inovação 06.txt - Notepad

File Edit Format View Help

Atividade: Criatividade e Inovação
Turma: 1
Estudante: Estudante 06

Segundo o autor, A inovação em uma empresa serve para gerar lucros e assegurar
a continuidade do negócio.

Criar um cliente.

Para Drucker uma organização inovadora significa que faz grandes investimento
em pesquisa, e tenha uma atitude voltada à inovação.

O capitalismo gera a destruição criativa e os ciclos econômicos, pois necessita
deles para manter-se vivo e em crescimento. Logo, todo o produto nasce com o
destino de morrer ou ser substituído por outro melhor. Assim, as empresas
precisam estar sempre atentas e preparadas para enfrentar o desaparecimento ou
substituição de seus produtos para que não acabem morrendo com eles.

O ciclo de vida de um produto conforme citado no texto se resume em:
1 - Os produtos têm vida limitada;
2 - As vendas dos produtos atravessam diferentes estágios, e cada um deles
reserva desafios e oportunidades diferentes para a empresa;
3 - Os lucros variam nos diferentes estágios do cliclo;
4 - Para cada estágio do ciclo é necessário desenvolve restratégias diferentes de
produção, vendas marketing, etc.

É o produto que entra no mercado, lançado pela empresa já tendo uma previsão
de ser superado por outro melhor depois de certo tempo, precisando assim ser
substituído.

Inovações Técnologicas se dividem em dois grupos, inovações de Produto e
inovações de processo, e Inovações não tecnologicas se subdividem em,
inovações de marketing, e inovações organizacionais.

As inovações incrementais são as market pull ou demand pull, que são as
inovações puxadas pelo mercado. As inovações radicais são as technology push ou
science push, inovações puxadas pela tecnologia ou pela ciência.

A inovação fechada é a que ocorre tradicionalmente dentro das empresas, e a
inovação fechada anda na contramão deste modelo, na inovação aberta, o
conhecimento gerado externamente à empresa é aproveitado, pesquisas e /ou
desenvolvimentos não têm de ser feitos internamente, a lucratividade se desvincula
do pioneirismo e passa a se relacionar com a capacidade de articulação dos
elementos internos e externos em favor da empresa.

A: são as inovações mais radicais. elas extrapolam as necessidades do consumidor
e dão origem a setores novos.
B: tamém são inovações radicais, mas são desenvolvidas em laboratório antes de
serem confrontadas com as necessidades do consumidor.
C: são inovações incrementais. Limitam-se às necessidades do consumidor e
representam uma ampliação na linha de produtos voltada aos consumidores atuais.

A microsoft vem perdendo espaço em vendas porque não conseguiu acompanhar

Figura 21: Produção Textual do Estudante 06, de maior resultado

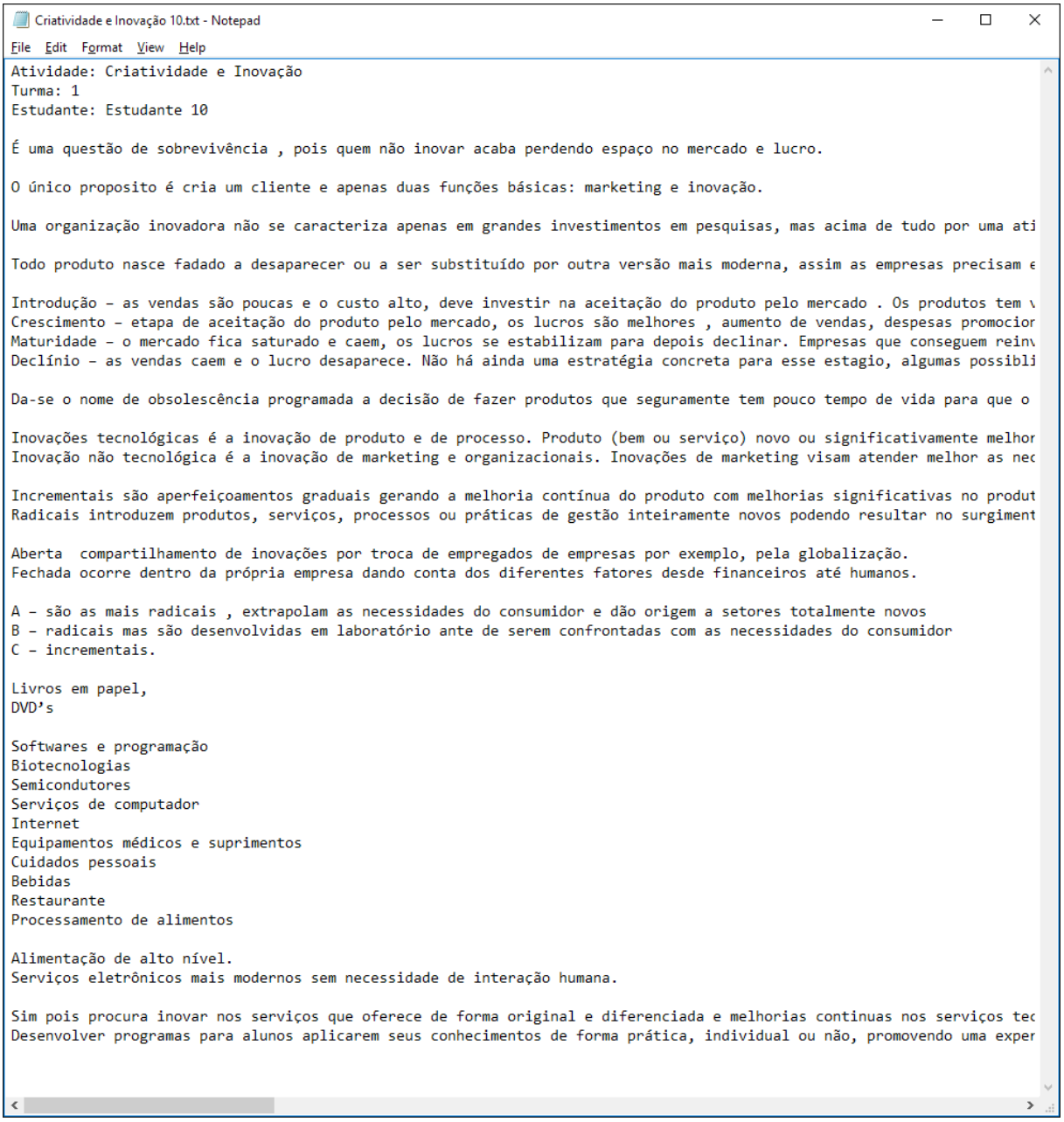


Figura 22: Produção Textual do Estudante 10, de menor resultado

Na Figura 23 pode-se observar a distribuição dos documentos analisados no espaço semântico. Cada item na legenda possui uma cor distinta para poder identificar o ponto no gráfico. Ao passar o mouse sobre o ponto, é possível visualizar o estudante e a nota correspondente.

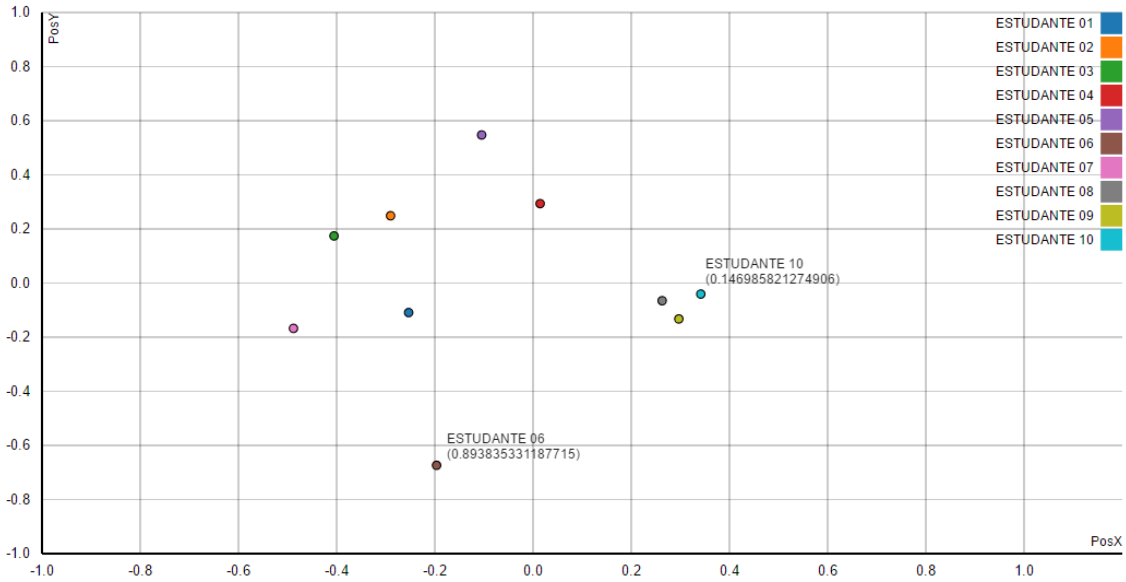


Figura 23: Visualização do resultado Informação e Decisão

4.5.3 Testes da Turma de Química Orgânica

Foram realizados os mesmos testes para averiguar o correto funcionamento do software com a turma de química orgânica. Após constatar que a ferramenta está funcionando como esperado, foi feita a análise das produções textuais de PEAD. Pode-se observar os resultados iniciais na Figura 24, onde é possível constatar que o ESTUDANTE-POLI 07 obteve o maior resultado da turma. O ESTUDANTE-POLI 03 obteve o menor resultado da turma.

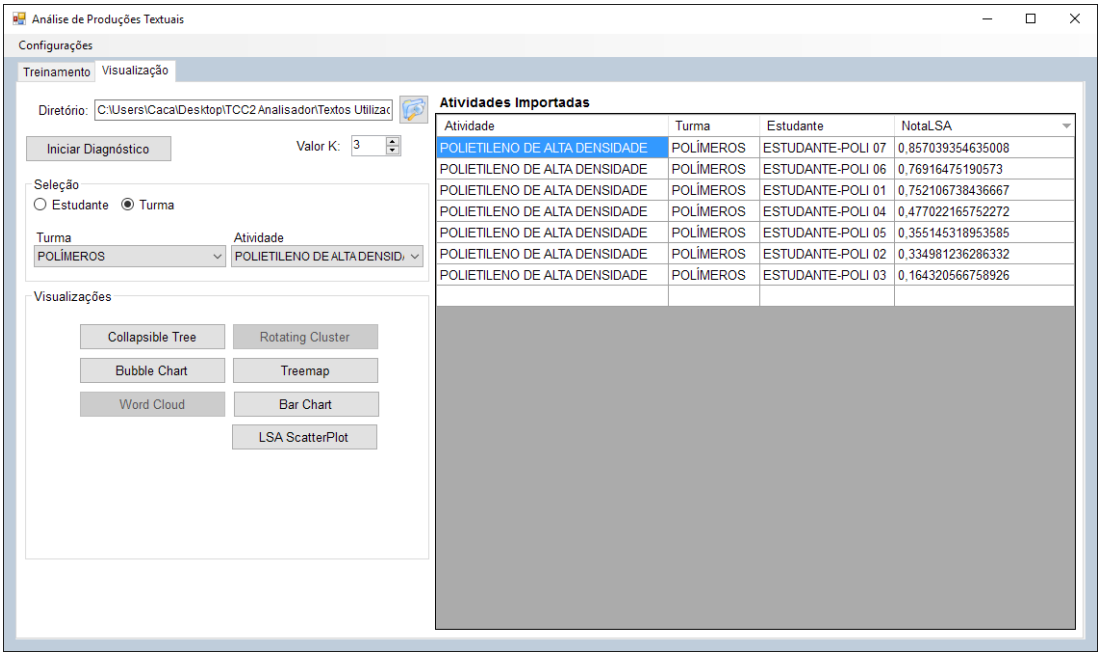


Figura 24: Análise da Turma de Polímeros

Na Figura 25 pode-se observar a distribuição das produções textuais no espaço semântico. Na Figura 26 evidencia-se a diferença na qualidade de produção entre os estudantes de maior e menor nota.

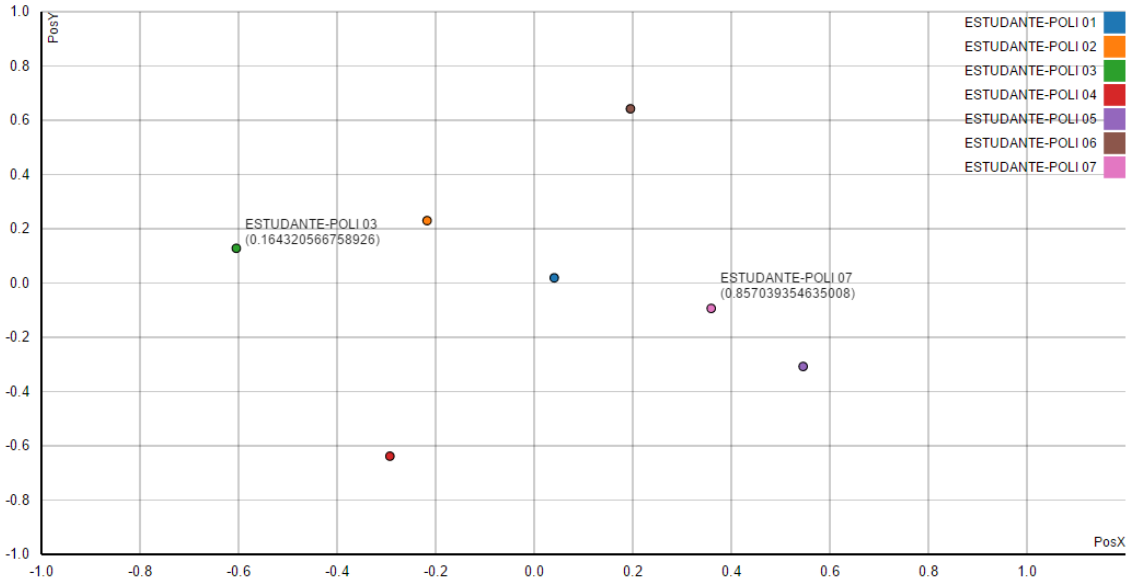


Figura 25: Visualização do Resultado de Polímeros

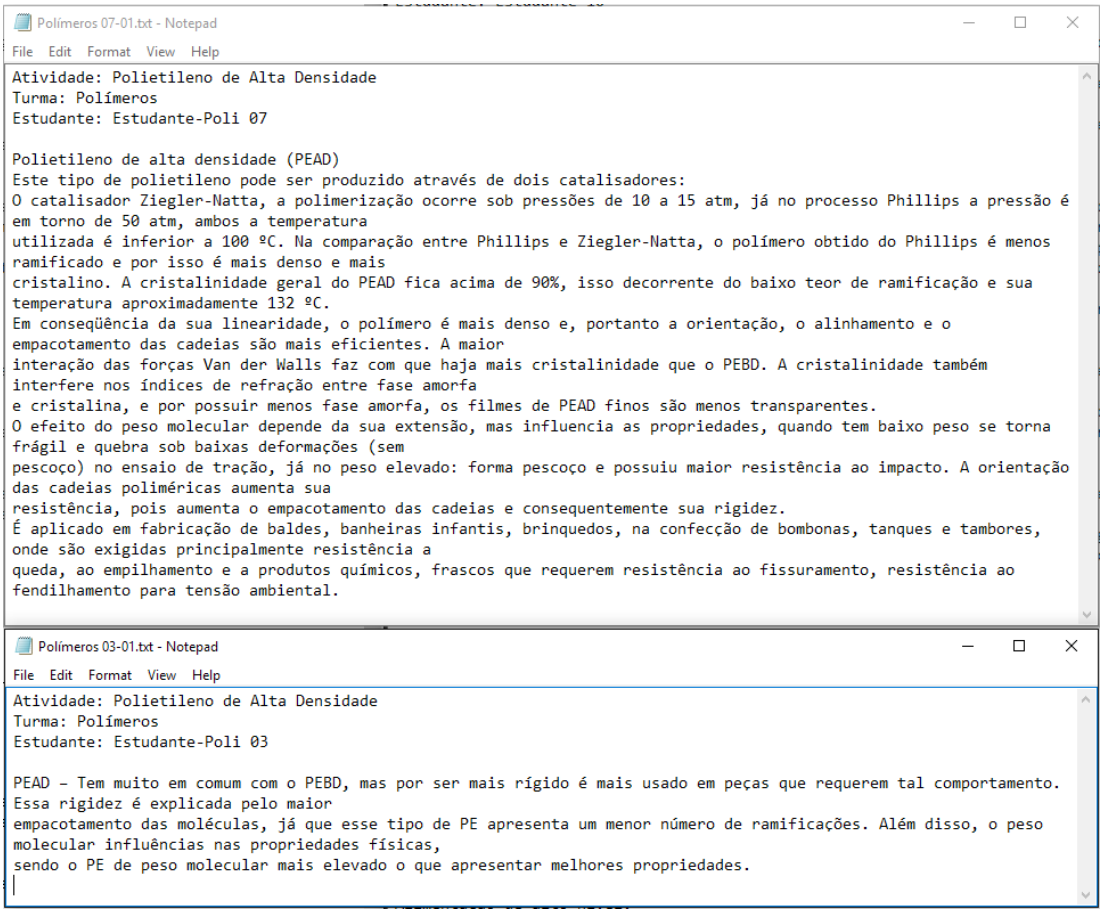


Figura 26: Comparação de Produções de Alunos de Polímeros

4.6 Avaliação dos Resultados

Os resultados obtidos foram satisfatórios, pois o sistema é capaz de avaliar a produção textual dos alunos e qualificá-los com referência ao texto utilizado no treinamento. A nota que o algoritmo atribui não deve ser utilizada como nota absoluta, uma vez que pode haver notas negativas no resultado final. No entanto, podem ser utilizadas para classificar as produções textuais entre si.

As matrizes geradas pela execução do algoritmo ficam disponíveis para serem utilizadas em outros escopos. Na visualização implementada, observa-se a distribuição dos documentos a partir da matriz V gerada na etapa de SVD. O processo de decomposição em valores singulares gera também a matriz U, que pode ser utilizada para mapear a distribuição dos termos no espaço semântico.

Desta forma, os objetivos do trabalho, com a implementação da análise LSA, foram atingidos, ficando a cargo do professor utilizar os resultados disponibilizados na avaliação de seus alunos. Uma avaliação mais completa será realizada no escopo do projeto de pesquisa, UnderMine Text Miner: Software de Mineração de Textos Educacionais, no qual este trabalho está inserido. Para isto, outras técnicas serão implementadas.

5 Conclusão

Este capítulo visa a apresentação de uma síntese das atividades desenvolvidas durante o desenvolvimento deste trabalho, descrevendo também os resultados obtidos e ideias para trabalhos futuros.

5.1 Síntese do Trabalho

A Mineração de Textos é um processo importante na extração do conhecimento, identificando palavras, expressões e termos relevantes em uma produção textual. No entanto, é necessário utilizar diversas técnicas de análise textual para poder extrair o máximo de conhecimento possível.

O principal objetivo deste trabalho foi o desenvolvimento de uma ferramenta para a análise de produções textuais no ambiente educacional a partir de uma nova implementação de um algoritmo de análise de semântica latente. Unindo o resultado deste algoritmo com a ferramenta de geração de visualizações utilizada por Lovato (2015), foi possível desenvolver uma nova funcionalidade ao software Under Mine Text Miner.

Para que o objetivo do trabalho fosse atingido, foram abordados assuntos sobre os conceitos da mineração de textos. Também foi abordado o processo de Análise de Semântica Latente, sendo apresentado um exemplo de funcionamento.

Com base no conhecimento adquirido em Mineração de Textos e Análise de Semântica Latente, foi possível o desenvolvimento de um sistema de análise de produções textuais. O funcionamento é dado da seguinte forma: é realizado o treinamento da ferramenta conforme texto base que os estudantes realizarão as atividades. A partir deste treinamento, onde o sistema possui todas as palavras e as respectivas frequências, as produções textuais dos alunos são processadas e analisadas. Neste momento, o sistema LSA calcula a distribuição e o peso das palavras presentes, utilizando estes dados para gerar as matrizes relevantes para efetuar a redução da dimensionalidade dos dados, a fim de realizar a comparação das produções com o texto de treinamento.

Com o sistema desenvolvido, foi realizada uma atividade com uma turma de Polímeros que serviu como amostra alvo do estudo, possibilitando a validação do sistema. A partir da análise dos dados fornecidos pelo sistema, foi possível concluir que a ferramenta é válida e agrega funcionalidades que aumentam a cognição do processo de Mineração de Textos.

5.2 Resultados e Contribuições

A área da Ciência da Computação está em constante evolução. Um profissional, para conseguir acompanhar estas mudanças, precisa estar sempre apto a lidar com novas áreas de estudo, buscando a resolução dos mais variados problemas computacionais. Desta forma, o esperado é que este profissional desenvolva a habilidade de pesquisa, para que possa sempre buscar conhecimento que possa aplicar em novas soluções, acompanhando o mercado e as necessidades dos usuários nas mais diversas áreas.

Ao iniciar este trabalho, foi proposto um projeto de aplicação prática, que visou estender a mineração de textos através da implementação da análise de semântica latente, a fim de oferecer novas técnicas de avaliação e qualificação de produções textuais. Para que fosse possível realizar este trabalho, foi necessário estender os estudos para além das áreas de conhecimento abordados no currículo do curso de graduação.

Durante os testes realizados, foi possível verificar que os resultados gerados a partir da LSA podem ser utilizados na qualificação das produções textuais, podendo ser utilizado para auxiliar o professor em suas funções de ensino. Verificou-se também que o sistema de visualização de dados foi flexível e adequado para o novo algoritmo implementado.

Conclui-se, assim, que o presente trabalho atingiu os objetivos especificados e deixou importantes contribuições para a mineração de textos no que diz respeito à extensão das funcionalidades disponíveis nesta área. Desta maneira, novos conceitos foram agregados aos já obtidos, contribuindo para o enriquecimento de mais uma área da Ciência da Computação.

5.3 Trabalhos Futuros

Este trabalho abordou a implementação de um sistema de análise de produções textuais no ambiente educacional, reutilizando algoritmos de Mineração de Texto, oferecendo novas funcionalidades, a fim de disponibilizar novas técnicas de análise textual.

Como sugestão para trabalhos futuros, seria de grande utilidade a disponibilização de uma interface gráfica *web*, oferecendo integração com plataformas como redes sociais, possibilitando novas fontes de informação.

6 Referências

ALMEIDA, M. B.; MARTINS, A. F. T. **Fast and Robust compressive summarization with dual decomposition and multi-task learning**. Proceedings of the Annual Meeting of the Association for Computer Linguistics. [S.l.]: [s.n.]. 2013.

ARANHA, C. N. **Uma Abordagem de Pré-Processamento Automático para Mineração de Textos em Português: Sob o Enfoque da Inteligência Computacional**. PUCRJ. [S.l.]. 2007.

BERRY, M. W.; DRMAC, Z.; JESSUP, E. R. **Matrices, Vector Spaces, andInformation Retrieval**. SIAM Rev. Philadelphia, PA. 1999.

CABRAL, M. O Texto Escrito. **Brasil Escola**, 2009. Disponivel em: <http://www.brasilescola.com/redacao/texto-escrito.html>. Acesso em: 30 Novembro 2015.

CHUANG, W. T.; YANG, J. **Extracting Sentence Segments for Text Summarization: A Machine Learning Approach**. Computer Science Department, UCLA. [S.l.]. 2000.

CORDEIRO, A. D. **Gerador Inteligente de Sistemas com Auto-aprendizagem para Gestão de Informações e Conhecimento**. UFSC, Departamento de Engenharia de Produção. [S.l.]. 2005.

CORREA, R.; LUDERMIR, T. Categorização Automática de Documentos: Estudo de Caso. **XVI Brazilian Symposium on Neural Networks**, 2002.

DEERWESTER, S. E. A. **Indexing by latent semantic analysis**. Journal of the American Society for Information Science. [S.l.]. 1990.

DELL. Text Mining (Big Data, Unstructured Data), 2015. Disponivel em: <documents.software.dell.com/Statistics/Textbook/Text-Mining>. Acesso em: 20 Novembro 2015.

EBECKEN, N. F. F.; LOPES, M. C. S.; COSTA, M. C. E. D. A. Mineração de Textos. In: REZENDE, S. O. **Sistemas Inteligentes:** Fundamentos e Aplicações. [S.l.]: [s.n.], 2003.

EIKVIL, L. **Information extraction from world wide web - a survey**. Norweigan Computing Center. [S.l.]. 1999.

EMMS, M.; LUZ, S. Machine Learning for Natural Language Processing. In: _____ **European Summer School of Logic, Language and Information**. [S.l.]: [s.n.], 2011.

GLYMOUR, C. et al. **Statistical themes and lessons for data mining**. Data Mining and Knowledge Discovery. [S.l.]. 1997.

GONG, Y.; LIU, B. **Creating Generic Text Summaries**. Document Analysis and Recognition, International Conference on. Los Alamitos, CA: [s.n.]. 2001.

GUPTA, V.; LEHAL, G. A Survey of Text Mining Techniques and Applications. In: _____ **Journal of Emerging Technologies in Web Intelligence**. [S.l.]: [s.n.], 2009.

JUÁREZ-GONZÁLEZ, A. et al. **Using Machine Learning and Text Mining in Question Answering**. Language Technologies Group, Computer Science Department, National Institute of Astrophysics, Optics and Electronics (INAOE), Mexico. [S.l.]. 2006.

JULIAN, K.; PEDERSEN, J.; CHEN, F. **A trainable document summarizer**. Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. [S.l.]: [s.n.]. 1995.

KLEMANN, M.; REATEGUI, E.; RAPKIEWICZ, C. **Análise de Ferramentas de Mineração de Textos para Apoio à Produção Textual**. [S.l.]. 2011.

LOH, S. **Abordagem Baseada em Conceitos para Descoberta de Conhecimento em Textos**. Instituto de Informatica UFRGS. [S.l.]. 2001.

LOVATO, G. **APLICAÇÃO DA MINERAÇÃO DE TEXTOS NA ANÁLISE DE PRODUÇÕES TEXTUAIS**. UCS. Caxias do Sul. 2015.

MACEDO, A. et al. **Using Text-Mining to Support the Evaluation of Texts Produced Collaboratively**. Education and Technology for a Better World; Selected papers of the 9th World Conference on Computers in Education. Bento Gonçalves: [s.n.]. 2009.

MARTINS, C. B. et al. **INTRODUÇÃO À SUMARIZAÇÃO AUTOMÁTICA**. Relatório Técnico RT-DC, 2. [S.l.]. 2001.

MORAIS, E.; AMBRÓSIO, A. **Mineração de Textos**. Instituto de Informática UFG. [S.l.]. 2007.

ORENGO, V. M.; HUYCK, C. **A stemming algorithm for the portuguese language**. [S.l.]: SPIRE. 2001. p. EIGHTH INTERNATIONAL SYMPOSIUM ON STRING PROCESSING AND INFORMATION RETRIEVAL.

PUFFINWARRELC, < *http://www.puffinwarellc.com/index.php/news-and-articles/articles/33-latent-semantic-analysis-tutorial.html?showall=1*>. Acesso em 30 Novembro 2015

SANGER, J.; FELDMAN, R. **The Text Mining Handbook**. [S.l.]: [s.n.], 2006.

SHEHATA, S.; KARRAY, F.; KAMEL, M. An Efficient Concept-Based Mining Model for Engancing Text Clustering. **IEEE Transactions On Knowledge And Data Engineering**, 2010.

SILVA, E. **Um sistema para extracao de informação em referências bibliograficas baseado em aprendizagem de maquina**. Centro de Informatica, UFPE. [S.l.]. 2004.

SOBEK. Sobek Mining. **Minerador de Textos Sobek**, 2015. Disponivel em: <http://sobek.ufrgs.br/index.html>. Acesso em: 30 Novembro 2015.

STEINBACH, M.; KARYPIS, G.; KUMAR, V. **A comparison of document clustering techniques**. Department of Computer Science and Egineering,. [S.l.]. 2000.

TAGCROWD. TagCrowd. **TagCrowd.**, 2015. Disponivel em: <http://tagcrowd.com>. Acesso em: 30 Novembro 2015.

TAN, Y.; WANG, J. **A support vector machine with a hybrid kernel and minimal Vapnik-Chervonenkis dimension**. [S.l.]. 2004.

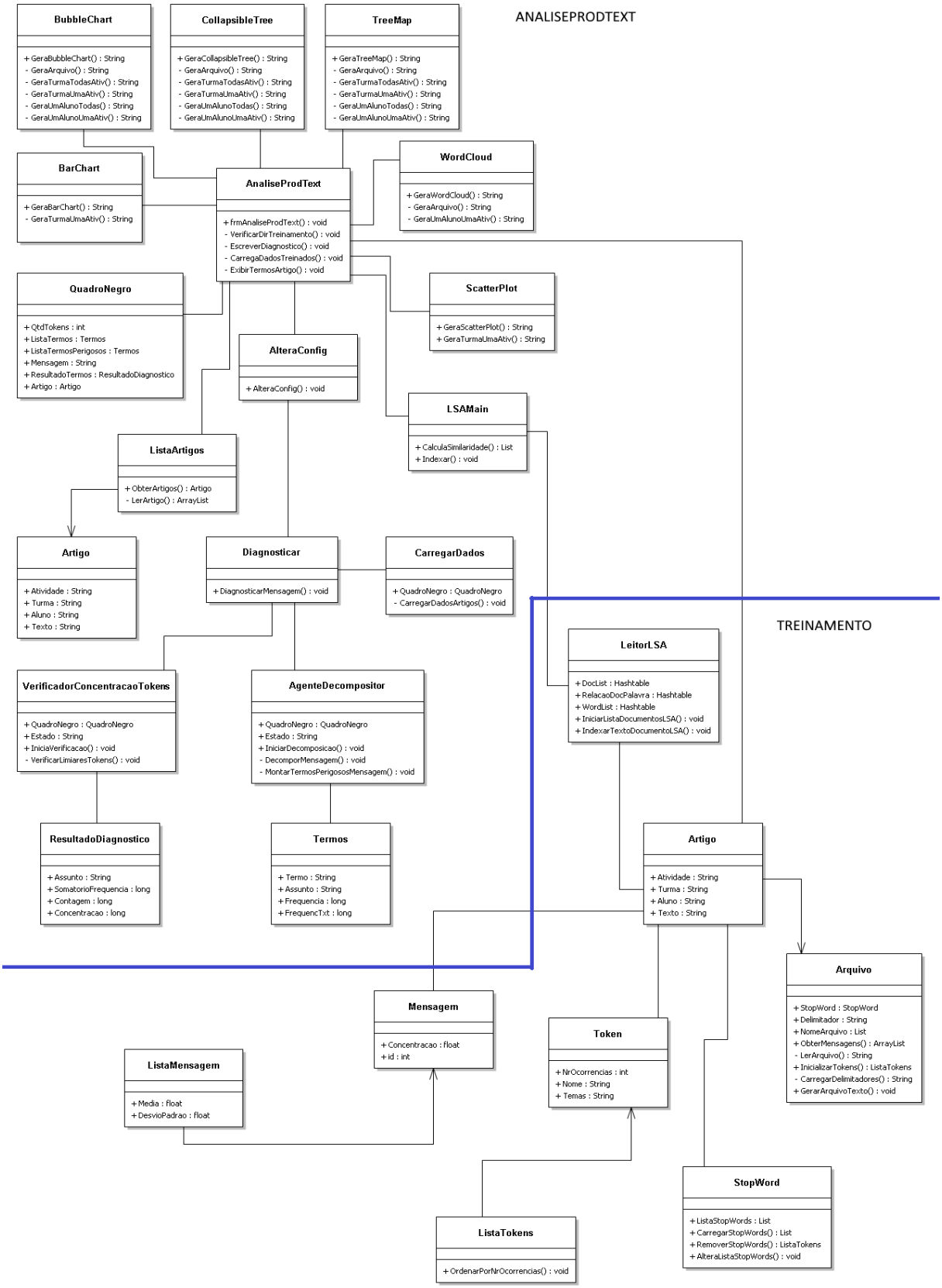
TEXTALYSER. TextAlyser. **TextAlyser.**, 2015. Disponivel em: <http://textalyser.net>. Acesso em: 30 Novembro 2015.

VASCONCELOS, L. M. R.; CARVALHO, C. L. **Aplicação de Regras de Associação para Mineração de Dados na Web**. Instituto de Informática Universidade Federal de Goiás. [S.l.]. 2004.

WONG, P.; WHITNEY, P.; THOMAS, J. **Visualizing Association Rules for Text Mining**. Pacific Northwest National Laboratory. [S.l.]. 1999.

WORDCOUNTER. Word Counter. **Word Counter**, 2015. Disponivel em: <http://www.wordcounter.net>. Acesso em: 30 Novembro 2015.

ANEXO A – DIAGRAMA DE CLASSE ANÁLISE DE PRODUÇÕES TEXTUAIS



ANEXO B – MANUAL DO PRODUTO

1 INSTALAÇÃO

Neste capítulo serão descritos os procedimentos necessários para que a ferramenta Análise de Produções Textuais seja utilizada. Também é descrito como pode-se importar o projeto na ferramenta de desenvolvimento, caso algum desenvolvedor tenha o interesse de alterar o sistema.

1.1 PARA DESENVOLVEDOR

Este capítulo é destinado para os desenvolvedores que desejam fazer manutenção no sistema. É descrito os softwares necessário e os passos para importar a aplicação na ferramenta de desenvolvimento.

1.1.1 Softwares necessários

Para a instalação da versão de desenvolvimento serão necessários os softwares abaixo:

a) Os programas liberados para o sistema Análise de Produções Textuais.
b) Visual Studio

   A versão atual do sistema Análise de Produções Textuais foi desenvolvida utilizando a versão 2013 do Visual Studio. Caso seja importado em alguma outra versão, podem haver algumas diferenças de pacotes que terão de ser ajustadas.

c) Apache HTTP Server

   Para a versão atual do sistema Análise de Produções Textuais foi utilizado o Apache HTTP Server 2.2. Os exemplos aqui contidos serão descritos utilizando esta versão. Mas nada impede de utilizar outro software para esta finalidade.

1.1.2 Instalação da Ferramenta

a) A instalação do Servidor Web Apache está descrita no capítulo 1.3.

   Porém, poderá ser utilizado outro servidor web.

b) O sistema Análise de Produções Textuais está disponibilizado em um arquivo zipado contendo os seguintes diretórios:
   - Implementação
   - Visualizações

c) Dentro do diretório *Implementação* estão os fontes do sistema. O arquivo que deve ser importado no Visual Studio é o (Implementação\AnaliseProdText\AnaliseProdText.sln).

d) Os diretórios contidos dentro de *Visualizações* possuem as implementações das visualizações que devem ser colocadas dentro do diretório do servidor web. Conforme os exemplos desde manual, ficaria conforme a seguir: C:\Webserver\Apache2.2\htdocs\. Abaixo deste diretório deve-se copiar os diretórios contidos em *Visualizações*.

1.2 PARA USUÁRIO

Este capítulo é destinado aos usuários da aplicação. Será descrito os softwares necessários e os procedimentos para instalar a aplicação no computador.

1.2.1 Softwares Necessários

Para a instalação da versão de desenvolvimento serão necessários os softwares abaixo:

a) Os programas liberados para o sistema Análise de Produções Textuais.

   Junto com os programas do sistema há um instalador do sistema, como o sistema foi desenvolvido utilizando a linguagem C#, só poderá ser instalado no sistema operacional Windows.

b) Apache HTTP Server

   Para a versão atual do sistema Análise de Produções Textuais foi utilizado o Apache HTTP Server 2.2. Os exemplos aqui contidos serão utilizando esta versão. Mas nada impede de utilizar outro software para esta finalidade.

1.2.2 Instalação da Ferramenta

a) A instalação do Servidor Web Apache está descrita no capítulo 1.3. Porém, poderá ser utilizado outro servidor web.
b) O sistema Análise de Produções Textuais está disponibilizado em um arquivo zipado contendo os seguintes diretórios:
   - Implementação
   - Visualizações
c) Dentro do diretório *Implementação\Instalador* está o instalador da aplicação. Executando o arquivo *setup.exe* a aplicação é instalada no computador.
d) Os diretórios contidos dentro de *Visualizações* possuem as implementações das visualizações que devem ser colocadas dentro do diretório do servidor web. Conforme os exemplos desde manual, ficaria conforme a seguir: C:\Webserver\Apache2.2\htdocs\. Abaixo deste diretório deve-se copiar os diretórios contidos em *Visualizações*.

1.3 INSTALAÇÃO SERVIDOR APACHE

O servidor apache deverá ser configurado conforme padrão. Apenas devese cuidar com o caminho que se disponibilizara os arquivos que ficarão publicados pelo servidor. Na instalação deste manual foi configurado o caminho (C:\Webserver\Apache2.2\htdocs). Neste caminho ficarão publicados os programas responsáveis pelas visualizações da aplicação.

A Figura 27 apresenta como ficará a estrutura de diretórios após configurado conforme escrito no capítulo 1.1.2 e 1.2.2.

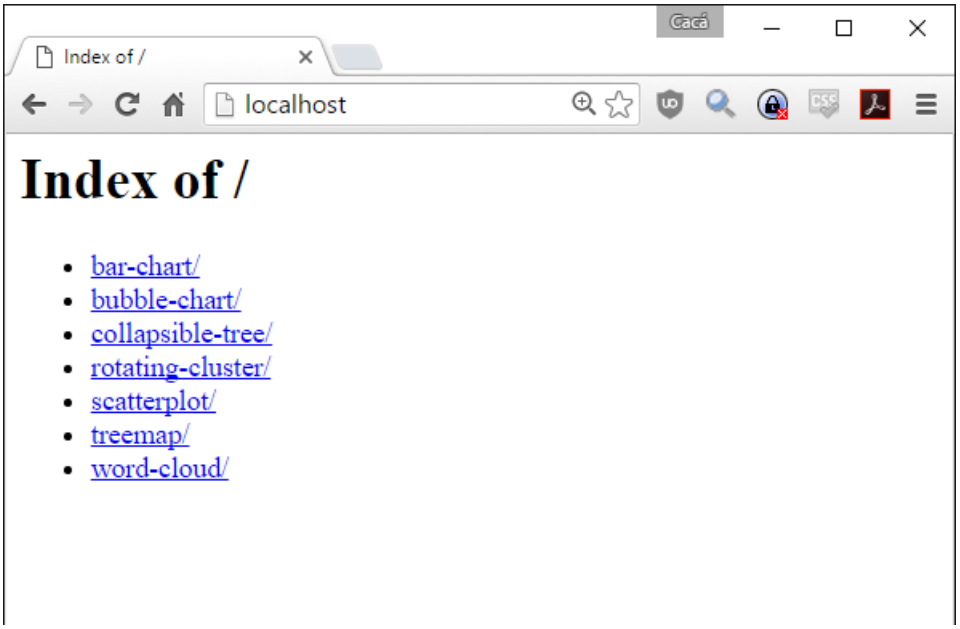


Figura 27 - Servidor Web.

2 UTILIZAÇÃO

Neste capítulo são descritas as funcionalidades do sistema Análise de Produções Textuais, configurações iniciais do uso e detalhamento de cada item da tela. Também são descritos como devem ser disponibilizados os textos para o treinamento da ferramenta e para a análise.

2.1 CONFIGURAÇÔES PARA PRIMEIRO USO

São necessárias duas configurações para o primeiro uso caso o servidor de web tenha sido instalado com alguma parametrização diferente do que o descrito neste manual. Estas configurações são necessárias devido a integração do sistema com as visualizações que serão apresentadas no Navegador padrão do computador.

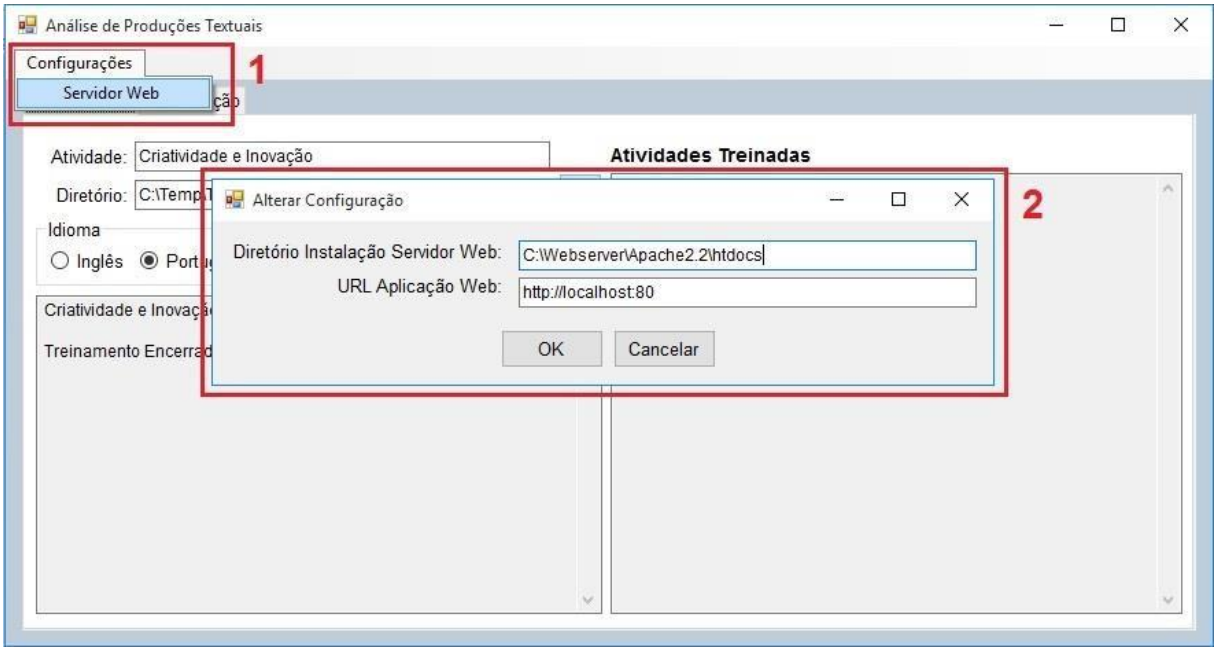


Figura 28: Configurações Análise de Produções Textuais

Conforme Figura 28, é necessário configurar o diretório de instalação do servidor web. O que está na imagem é o caminho padrão (C:\Webserver\Apache2.2\htdocs). Também é necessário configurar a URL da aplicação web. Por padrão é configurado o caminho (http://localhost:80), porta 80 instalada por padrão na instalação do servidor Apache. Após a configuração inicial, o sistema instalado sempre iniciará com o que foi salvo.

2.2 TREINAMENTO

Descrição das funcionalidades da tela:

a) Atividade: É a descrição da atividade que será treinado o sistema. Esta descrição é importante, porque os textos analisados posteriormente deverão conter o respectivo assunto ao serem importados.
b) Diretório: Deverá ser informado o diretório que contém os artigos em formado *txt* que serão importados no treinamento.
c) Idioma: O sistema está preparado para aceitar textos no idioma Inglês e Português. Importante informar o idioma correto, pois os textos analisados deverão estar no mesmo idioma.
d) Processar Artigos: Irá importar os artigos conforme o diretório informado para a importação. Esta ação irá criar um diretório (C:\listasTermos) caso seja a primeira vez que o sistema seja treinado. Este diretório armazenará os treinamentos.
e) Abaixo do Idioma é apresentada uma listagem com os artigos importados no treinamento que está sendo realizado, assim como uma mensagem informando que o treinamento foi efetuado.
f) Atividades Treinadas: Apresenta uma listagem das atividades que já foram treinadas no sistema.

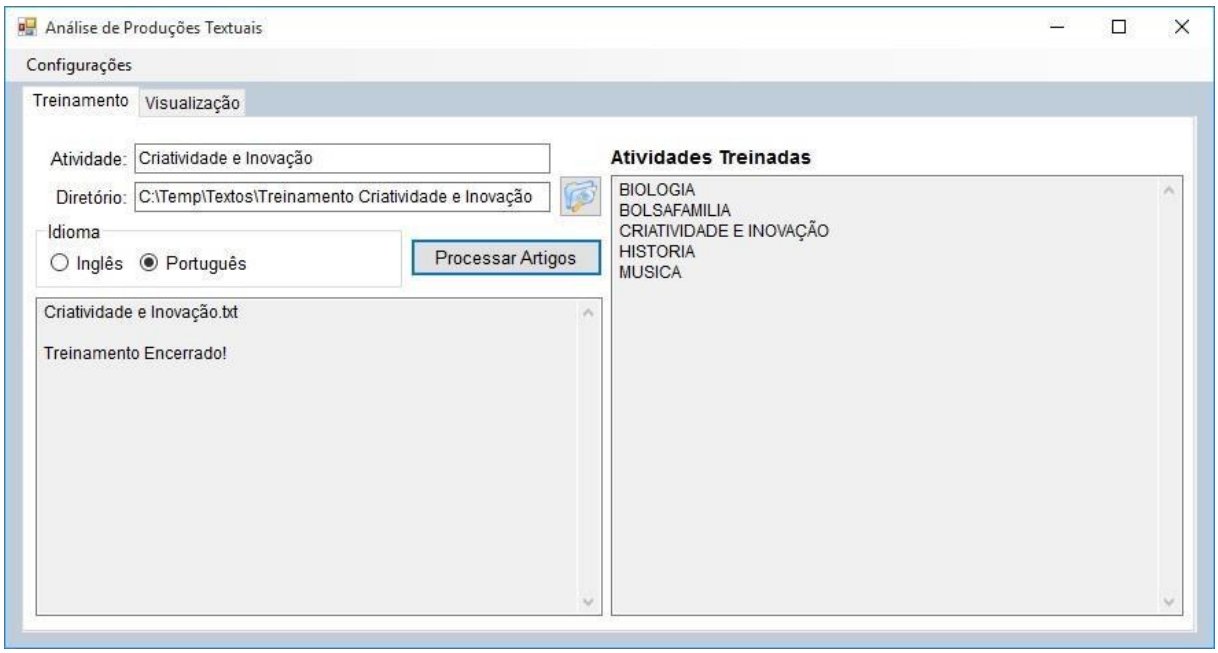


Figura 29: Treinamento Análise de Produções Textuais

O treinamento de uma atividade não é incremental, ou seja, sempre que se deseja adicionar um texto a uma atividade já treinada a fim de aumentar o nível de conhecimento do sistema sobre determinada atividade, deve-se treinar novamente com todos os textos desta atividade.

2.2.1 Arquivos de Entrada

Tanto os arquivos de entrada para o treinamento quanto para a análise têm que estar no formato TXT e tem que estar salvos utilizando o padrão UTF8.

A Figura 30 representa um arquivo de treinamento sendo salvo conforme os formatos corretos.

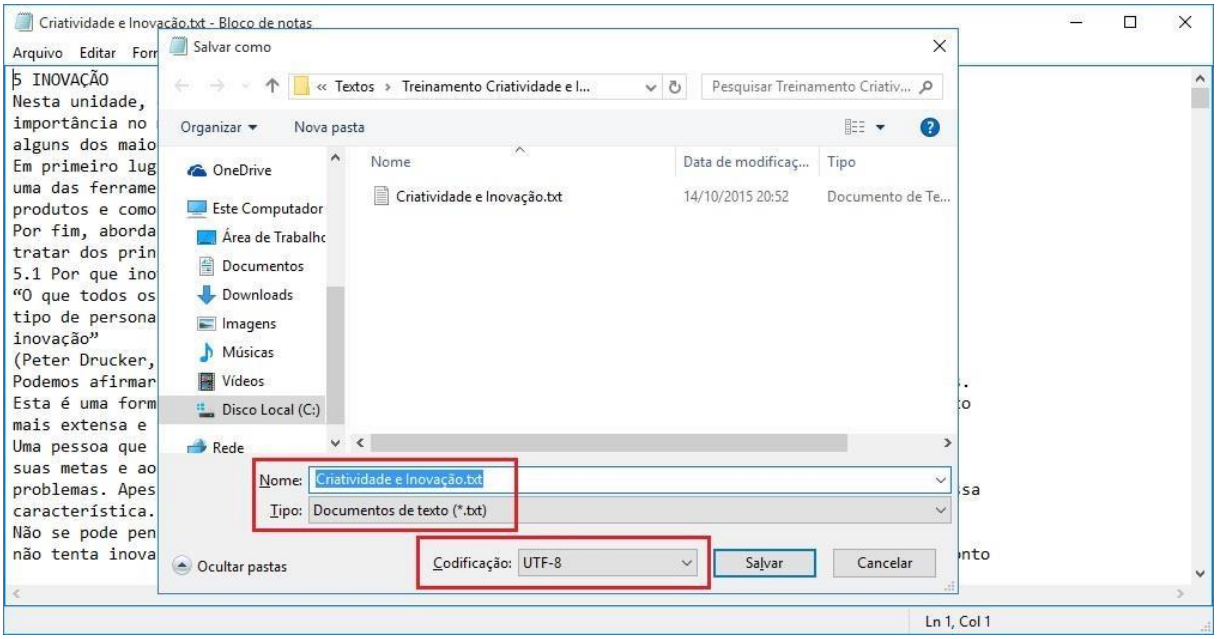


Figura 30: Arquivo Treinamento.

2.3 ANÁLISE

Descrição das funcionalidades da tela:

a) Diretório: Deverá ser informado o diretório que contém os artigos em formado *txt* que serão importados para análise.
b) Iniciar Diagnóstico: Irá importar os artigos conforme o diretório informado para a importação. Após esta ação o sistema está pronto para ser manipulado com o objetivo de analisar os textos importados.

c) Atividades Importadas: Apresenta um resumo dos textos que foram importados, contendo informações sobre a atividade, a turma e os estudante de cada artigo.

d) Seleção: A combinação dos textos para análise é representada na Figura 32. Pode-se selecionar para visualização o texto de um estudante em todas as atividades ou em uma atividade. Ou pode-se selecionar os textos de uma turma inteira em uma atividade ou em todas as atividades.

e) Visualizações: Cada botão representa uma visualização. As visualizações abrirão no Navegador padrão do computador. As visualizações estão disponíveis conforme seleção abaixo:

- Collapsible Tree – Todas as seleções.
- Bubble Chart – Todas as seleções.
- Word Cloud – Apenas um estudante em uma atividade.
- Rotating Cluster – Todas as seleções.
- Treemap – Todas as seleções.
- Bar Chart – Apenas uma turma em uma atividade.
- LSA ScatterPlot – Apenas uma turma em uma atividade

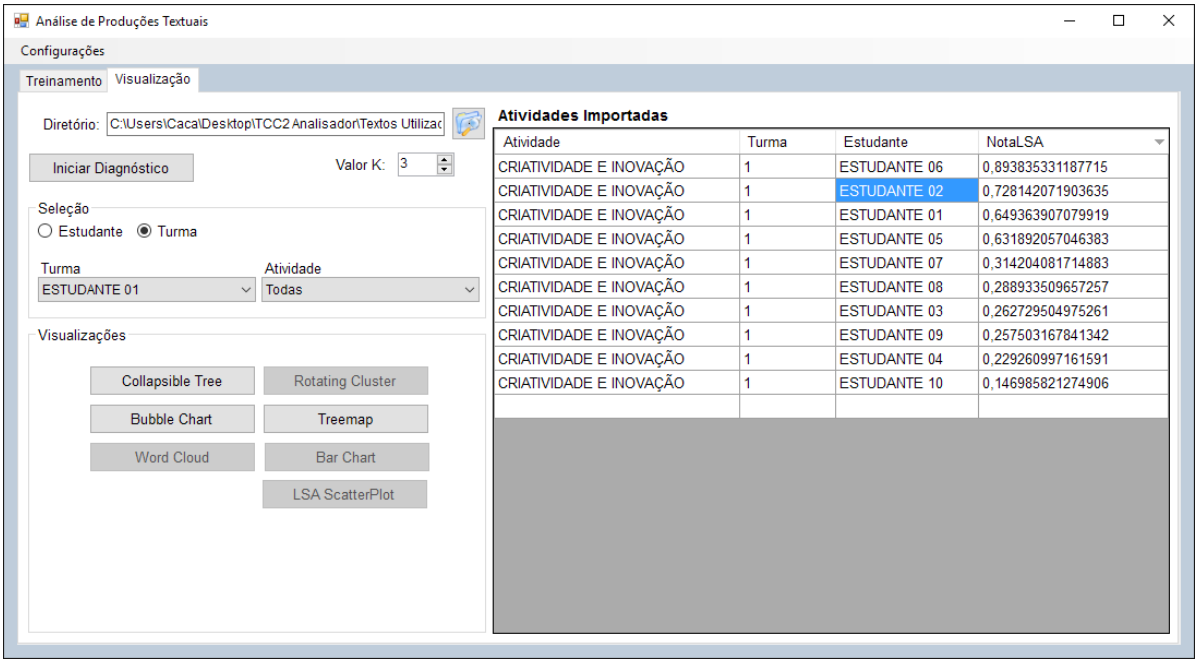


Figura 31: Análise de Produções Textuais.

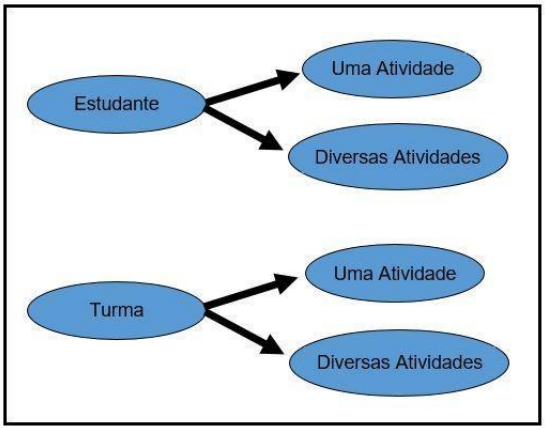


Figura 32: Combinação para seleção de textos para análise.

2.3.1 Arquivos de Entrada

Os textos para análise, além de estarem salvos em TXT e no formato UTF8 conforme imagem da Figura 30, devem possuir um cabeçalho que os identifique. O cabeçalho deve ser composto conforme abaixo:

Atividade: Criatividade e Inovação

Turma: 1

Estudante: Estudante 01

Importante verificar que a atividade precisa ter exatamente o mesmo nome da atividade que foi colocada no treinamento conforme Figura 29, inclusive com a mesma acentuação, caso utilizado. Um exemplo de cabeçalho e arquivo de entrada para análise é demonstrado na Figura 33.

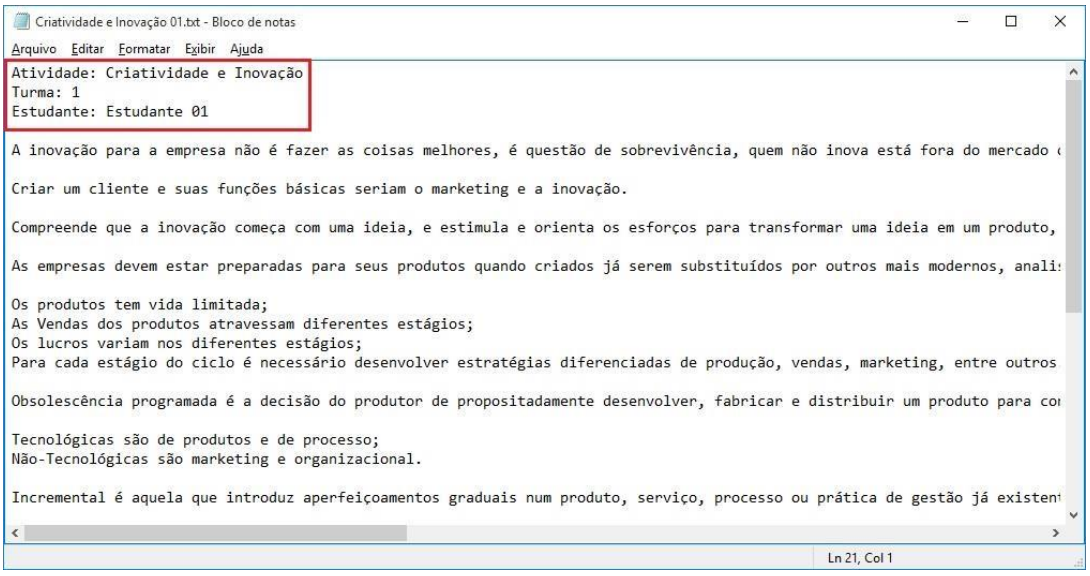


Figura 33: Arquivo para análise.