

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO  
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA



UFES

GERAÇÃO AUTOMÁTICA DE *RUBRICS*  
ATRAVÉS DA IDENTIFICAÇÃO DE  
INFORMAÇÕES-CHAVE EM RESPOSTAS  
DISCURSIVAS

MARCOS ALÉCIO SPALENZA

VITÓRIA-ES

2017

## *Resumo*

O processo de avaliação é muito importante para a verificação de aprendizagem dos estudantes e manutenção do andamento do ensino conforme o currículo previsto. A avaliação formativa garante ao professor uma visão geral do ensino, medindo a efetividade do aprendizado. Porém, conforme é ampliado o acesso à educação, a aplicação de métodos avaliativos se torna ainda mais necessária e a carência de apoio em sua aplicação é considerada como um fator limitante.

O suporte computacional dos métodos pedagógicos visa melhorar a qualidade dos materiais para impactar diretamente a aprendizagem. Neste trabalho, apresentamos uma ferramenta de apoio à correção de respostas discursivas curtas segundo os termos e sua representatividade por classe de nota. A adoção da seleção de características proporciona a criação de modelos de avaliação mais próximos ao critério do professor. Com os trechos de respostas identificados, esperamos padrões que tornem o sistema um avaliador em potencial. Assim, a ferramenta proposta para modelagem do método avaliativo e *feedback* para suporte ao ensino é aqui chamada de mapas de características.

Para desenvolvimento deste projeto coletamos cinco bases de dados: duas nacionais e três estrangeiras. Os experimentos iniciais buscavam analisar a interpretação da base de dados pelo sistema para modelar o critério avaliativo, com a finalidade de identificar o padrão de classificação do professor. Em duas das bases de dados testadas, de disciplinas da Universidade Federal do Espírito Santo e da Universidade do Norte do Texas, o resultado obtido foi de uma classificação equivalente após a redução de dimensionalidade com respectivamente 24,84% e 11,48% da informação.

**Palavras-chave:** Mapa de Características, Avaliação Assistida por Computador, Mineração de Dados Educacionais, Aprendizado Semiautomático, Extração de Informação

---

## Sumário

<b>1</b>	<b>Introdução</b>	<b>4</b>
<b>2</b>	<b>Trabalhos Relacionados</b>	<b>5</b>
<b>3</b>	<b>Problemática</b>	<b>8</b>
<b>4</b>	<b>Objetivos</b>	<b>11</b>
<b>5</b>	<b>Metodologia</b>	<b>13</b>
<b>6</b>	<b>Bases de Dados e Experimentos Iniciais</b>	<b>16</b>
6.1	Base de Dados do Vestibular UFES . . . . .	16
6.2	Base de Dados das Disciplinas UFES . . . . .	17
6.3	Base de dados da Universidade do Norte do Texas (Inglês) . . . . .	17
6.4	Base de Dados do Concurso ASAG-SAS no <i>Kaggle</i> ( <b>Inglês</b> ) . . . . .	17
6.5	Base de Dados da <i>UK Open University</i> . . . . .	18
6.6	Experimento: Conformidade com a Avaliação Humana . . . . .	18
6.7	Experimento: Relação Termo-Nota . . . . .	19
6.8	Experimento: Conteúdo-Chave para Avaliação . . . . .	21
<b>7</b>	<b>Cronograma</b>	<b>24</b>

## 1 Introdução

As avaliações de aprendizado são fundamentais para o ensino ao apontar o desempenho da turma com o progresso nos conteúdos. Com aplicações frequentes, as atividades permitem ao professor interagir com os alunos e com os materiais pedagógicos para reformulação e aperfeiçoamento da sua metodologia. Desse modo, é com o acompanhamento da disciplina e o apoio ao educando que as atividades formativas permitem a reformulação do processo de ensino-aprendizagem (BARREIRA; BOAVIDA; ARAÚJO, 2006). Por meio das atividades podemos identificar o domínio dos estudantes sobre o contexto e sua capacidade de realizar inferências sobre o assunto. O papel da avaliação, portanto, é diagnosticar, apreciar e verificar a proficiência dos alunos para que o professor atue no processo de formação de modo a consolidar o aprendizado (OLIVEIRA; SANTOS, 2005).

Ao observar problemas no modelo de ensino-aprendizagem, o professor pode agir de forma a contorná-los, personalizando a estrutura curricular. É através das atividades, portanto, que observamos paralelos do conhecimento individual dos alunos. Um modo de aperfeiçoar a aplicação das atividades em quantidade e qualidade é dada através da mediação tecnológica. A mediação tecnológica na criação, avaliação, recomendação e visualização em dados educacionais apoia o professor na melhoria e no acompanhamento do currículo do aluno (PAIVA et al., 2012). É com as ferramentas de apoio, então, o tutor pode verificar a aptidão dos estudantes, de forma individual ou coletiva, para adequá-los à disciplina.

Sabendo que existe um critério de correção para respostas discursivas, propomos uma abordagem de reconhecimento do padrão de correção. Neste projeto então descrevemos uma abordagem para reconhecimento de padrões textuais de acordo com a classificação da base de dados. Considerando o modelo de notas atribuído como classes, observamos grupos de respostas e os termos referenciais. Com essa modelagem do critério por classe, então, esperamos demonstrar o método avaliativo com a criação de *feedbacks* como o quadro de *rubrics* (ARTER; CHAPPUIS, 2006) e, consequentemente, melhorar os métodos de avaliação automática (SPALENZA et al., 2016).

Este projeto é apresentado em 7 seções. Na Seção 2 apresentamos trabalhos correlatos recentes da área, sobre a extração de informação e avaliação de textos educacionais. Na

Seção 3 descrevemos alguns problemas comuns encontrados na literatura para, na Seção 4, apresentarmos os objetivos do projeto. Na Seção 5 descrevemos o escopo do sistema desde o protótipo inicial aos testes com algoritmos. Na Seção 6 são listadas as bases de dados coletadas e os resultados obtidos no protótipo inicial. Por fim, na Seção 7, apresentamos o cronograma esperado para o desenvolvimento dessa pesquisa.

## 2 Trabalhos Relacionados

Segundo Pérez-Marín, Pascual-Nieto e Rodríguez (2009), apoiando-se na evolução das técnicas de inteligência computacional, a Avaliação Assistida por Computadores evoluiu muito na última década. Estudos passaram da análise simples do modelo de gradação da nota para interpretação do conteúdo e da metodologia como um todo. Então, a partir dessa evolução, os sistemas começaram a reconhecer padrões para modelar o critério avaliativo do professor nas atividades.

O estudo realizado por Butcher e Jordan (2010) é resultado da comparação entre os sumários elaborados por especialistas e métodos automáticos baseados em linguística computacional ou em extração de palavras-chave. Para estabelecer essa comparação foram utilizados seis especialistas para avaliarem de forma binária (correto / incorreto) cada questão. Os sistemas comparados foram o *Intelligent Assessment Technologies FreeText Author* para linguística computacional, o *OpenMark* e expressões regulares para extração de palavras-chave.

Segundo essa pesquisa o sistema avaliado, o *FreeText Author*, têm resultados equivalentes à análise dos especialistas. Quando comparado com os outros sistemas, há uma proximidade entre resultados. A conformidade da ferramenta com os especialistas foi acima de 90% de acurácia na marcação, com desempenho superior aos das ferramentas de extração de palavras-chave. A abordagem porém só leva em consideração níveis de classe binários, sendo a avaliação por classe de nota um problema mais complexo.

Voltado diretamente para verificação do sistema automático para avaliação de respostas discursivas curtas, Mohler, Bunescu e Mihalcea (2011) abordam o tema com análise pela

similaridade das respostas com apoio de grafos de dependência textual. A proposta deste artigo, portanto, foi a extração do contexto dos termos para melhoria do processo de avaliação. A resposta candidata passada pelo especialista foi utilizada como base para extração de características e reconhecimento de padrões de escrita através da similaridade dos documentos e da interpretação sistêmica dos grafos de dependência morfossintática. A comparação direta com a resposta candidata, pré-estabelecida como correta, foi realizada segundo oito medidas de similaridade semântica. A associação dos dois módulos, a análise documental por similaridade e o grafo de dependências, gera o padrão de correção do sistema.

No trabalho a base de dados utilizada para os testes foi coletada na Universidade do Norte do Texas na disciplina de Estrutura de Dados. O *Texas Dataset*, como foi chamado pelos autores, é descrito com mais detalhes na Seção 6. Nesse *dataset*, os avaliadores humanos coincidiram na correção de 0 a 5 pontos em apenas 57,7% das notas e em 80,6% se fossem desconsiderados erros de até 1 ponto de divergência na avaliação. Os resultados obtidos na classificação pelo sistema foram de 85% de *precision* e 62% de *recall*, estratificado em *12-fold cross validation*. A raiz do erro médio quadrático apresentado foi de 0,98 pontos, sendo de 0,86 pontos quando computado por questão. A correlação alcançada entre as respostas para essa base de dados foi de 0,480.

A metodologia proposta por Gomaa e Fahmy (2012) aplica análise de similaridade de texto para desenvolver avaliações mais próximas das respostas candidatas. Aplicado ao *Texas Dataset*, o sistema utiliza métricas complementares por termos e por *corpus*. Na primeira metodologia, de análise morfológica por termos, foram 13 métodos distintos aplicados entre a comparação de caracteres e as métricas de similaridade. Na segunda foram dois métodos: distribuição de palavras similares e co-ocorrência (DISCO), computando a frequência *I) ordenada pela similaridade sobre duas palavras em colocação (correferência) no corpus e II) ordenada pela similaridade sobre duas palavras em termos similares*. De forma não supervisionada, este estudo alcançou resultados de correlação de 0,504 acima do *baseline* (MOHLER; BUNESCU; MIHALCEA, 2011), identificando maior tendência nas respostas das questões.

Um algoritmo distinto para o processo de comparação com respostas candidatas é resultado do estudo realizado por Noorbehbahani e Kardan (2011). Neste artigo a resposta

apresentada pelo professor é comparada com o *dataset* através do módulo de avaliação de sumários *ROUGE*, do método de extração de contexto *Latent Semantic Analysis - LSA*, do algoritmo de comparação entre respostas *ERB*, do sistema de avaliação de traduções *BLEU* e do cálculo de co-ocorrência de *n-grams*. Foram adicionados ainda uma modificação do algoritmo *BLEU* para abordagens de correção para respostas discursivas e uma ponderação entre *n-grams* e o *LSA*. O *dataset* foi construído através de um AVA, coletando 45 questões e 300 respostas. Como resultado o artigo apresentou a covariância entre os métodos e a resposta candidata, sendo o melhor desempenho o do *BLEU* modificado com 0,85. A correlação medida foi dada pela covariância entre a similaridade das respostas e a nota do especialista pelo desvio padrão de ambos.

No artigo de Ramachandran, Cheng e Foltz (2015), os autores mostram uma abordagem relacionada com a análise dos documentos de resposta dos alunos. O objetivo deste estudo foi criar de forma sistêmica a análise das respostas contornando a rigidez de uma única chave de correção. Neste trabalho os autores apresentam modelos criados a partir da sumarização das respostas bem avaliadas. Assim os sumários de classe são aguardados como correlatos ao modelo avaliativo do professor para a nota máxima. O método utilizado é a coesão por análise de grafos, similaridade e clusterização. As respostas mais representativas são extraídas conforme maior seja sua cobertura semântica *intracluster*.

Em um trabalho complementar, Ramachandran e Foltz (2015) propõe uma forma de geração de respostas candidatas analisando os documentos avaliadas com nota máxima, o quadro de *rubrics* e, se existir, o material de apoio. O modelo extrai relações entre conceitos do conteúdo retornando um conjunto de padrões de forma não ordenada. Os testes realizados mostram resultados equiparáveis com os do vencedor do concurso do *Automatic Short Answer Grading - ASAP* do *Kaggle*<sup>1</sup> substituindo as expressões regulares lá criadas manualmente. Outra base de dados testada, o *Texas Dataset*, a raiz do erro médio quadrático foi reduzida de 0,98 para 0,86 pontos e de 0,86 para 0,77 pontos na análise por questão.

Outra técnica similar para análise do conteúdo em documentos dos alunos foi proposto por Oliveira, Ciarelli e Oliveira (2013). O objetivo deste é entregar ao estudante recomendações de classes de atividades como *feedback* conforme o exercício submetido. Três bases de

---

<sup>1</sup> Kaggle <<https://www.kaggle.com>>

dados foram utilizadas, duas jornalísticas e uma de exercícios de programação. De acordo com o conhecimento do especialista na classificação/avaliação dos dados, a ferramenta enviou possíveis melhorias no aprendizado segundo o conteúdo. Os resultados de *precision* apresentados com para tais dados foram de 89%.

### 3 Problemática

Na literatura da Avaliação Assistida por Computadores encontramos alguns problemas do tópico de avaliação automática de questões discursivas que temos interesse em trabalhar neste projeto. Apesar de ser um estudo realizado há décadas, na avaliação automática de questões discursivas encontram-se desafios pouco abordados e dados como problemas do processo de avaliação pela máquina. Nos primeiros sistemas, a modelagem de questões discursivas era um trabalho realizado com o texto bruto. A partir disso, a busca por equivalência entre a resposta esperada e o texto dos estudantes falhou por inúmeras vezes na padronização dos documentos e na identificação de sinônimos (LEFFA, 2003). O estudo dessa problemática derivou discussões em torno da identificação do conhecimento nos documentos escritos pelo aluno, realizada atualmente em boa parte dos algoritmos. Técnicas de Aprendizado de Máquina, Estatística, Processamento de Linguagem Natural, Expressões Regulares e Reconhecimento de Padrões, dentre outras, foram testadas para a recuperação dos diferentes modelos de resposta e avaliação.

Na revisão da literatura de avaliação automática de questões discursivas produzida por Burrows, Gurevych e Stein (2015), os autores reúnem 37 trabalhos realizados na área. Durante essa revisão, o autor destaca o problema da profundidade do aprendizado, em tradução literal de “*depth of learning*”, separando as atividades em dois grupos: de reconhecimento e de recuperação. Tais modelos têm diferentes intúitos na aquisição de informação do aluno o que gera diferentes modos de processamento. No Brasil, conhecemos essas por questões abertas e fechadas, nomenclaturas também citadas pelos autores. Essa divisão estabelece a diferença entre as atividades que exploram apenas a necessidade de identificação e organização de conteúdo e as que dependem de construção de ideias visando respostas próprias e originais. Definimos, então, a liberdade do aluno na criação do seu conjunto de resposta



como a chave para separar atividades que necessitem de maior ou menor conhecimento factual, ou respectivamente, questões abertas ou fechadas.

Um problema com esse viés é reagir às questões discursivas factuais e opinativas da forma adequada (BAILEY; MEURERS, 2008). É esperado que o sistema lide com a liberdade de escrita do aluno recuperando o conteúdo. A forma aqui proposta para contornar esse problema de identificação do critério avaliativo parte de interações com o especialista. Essas interações são requisições de correção para buscar avaliações específicas de padrões textuais das respostas. Esse processo seleciona documentos que indiquem certo grau de distinção por grupo, resultante de uma clusterização inicial (OLIVEIRA et al., 2014). Coletamos assim a classificação dada para os itens elencados como relevantes pelo sistema em cada *cluster* para continuidade do processo avaliativo.

Desta forma, a modelagem semiautomática do processo permite que encontremos subgrupos de respostas por *cluster*. A partir daí devemos encontrar qual é o critério avaliativo do professor, sem precisar de uma chave de resposta, dado até o momento como necessário (BUTCHER; JORDAN, 2010; MOHLER; BUNESCU; MIHALCEA, 2011; RAMACHANDRAN; CHENG; FOLTZ, 2015). Alguns procedimentos de extração de informação e análise de padrões aqui propostos na Seção 5 visam remontar o critério do professor e otimizando o processo de correção automática.

O problema passa da necessidade de avaliação para o aumento do número de padrões à serem avaliados pelo professor para relacionar conteúdo com classificação. Butcher e Jordan (2010) citam alguns problemas no processo avaliativo automático do *FreeText Author*. O primeiro, a omissão do padrão de avaliação, já foi discutido acima. O segundo é a identificação de palavras incorretas associando-as com sua forma correta. O terceiro problema, conforme os autores, é a necessidade de identificação estrutural da sentença. A quarta dificuldade listada é o tratamento de classificações incorretas do especialista. O quinto problema é o conflito de um padrão correto de resposta com uma avaliação dada como incorreta. A relação entre a quarta e a quinta dificuldade listada é apresentada pelos autores como um sexto problema, dada pela inconsistência no módulo de interpretação textual do sistema, afetando diretamente a sua confiabilidade como avaliador.

Os problemas linguísticos listados por aquele autor (2º e 3º) de reconhecimento estruturais e semânticos ainda não são trabalhados de forma explícita na nossa proposta. A ferramenta alvo deste projeto faz análises diretas na frequência de ocorrência das palavras e realiza pré-processamentos para tornar grupos de termos e documentos equivalentes. Segundo Ramachandran, Cheng e Foltz (2015), a avaliação com base na sintaxe, ortografia ou organização não são suficientes para as questões discursivas curtas. Tendo isso em vista, o protótipo do sistema visa expandir a implementação de módulos linguísticos independentes, tal como o *stemming*, a remoção de *stopwords* e a separação por *n-grams* já testadas em português e inglês.

Após os módulos linguísticos, os demais problemas serão cautelosamente trabalhados através de métodos de recuperação da informação. Assim, o 4º problema esclarece a dificuldade em separar possíveis *outliers* das informações relevantes que foram apresentadas em uma única ou poucas ocasiões no banco de dados. Nosso método de contornar essa adversidade é dada pelo envio de padrões distintos para a avaliação humana, em busca de referências ao conteúdo. Desta forma, a ferramenta trabalha cada nota como um rótulo dado pelo especialista, criando um modelo mais detalhista do que a análise de uma resposta candidata elaborada pelo professor.

No 5º problema, sobre a localização de padrões de avaliação incorretamente classificados pelo especialista, há um impacto grave na adaptabilidade da máquina como avaliador. Esse tópico ainda não é trabalhado no protótipo, ao qual esperamos adequá-lo para interpretar alterações na avaliação para ajustar seu modelo de classificação, extraindo cada alteração como conhecimento. Assim, o sistema poderá identificar qual informação difere na resposta para o modelo avaliativo construído. Tal processo torna-o mais específico ou genérico segundo a variação da nota atribuída ao documento. O mesmo esperamos que ocorra no último problema listado pelos autores, a confiabilidade do sistema na modelagem do critério de avaliação. Com o ajuste da avaliação pelo professor durante etapas de revisão, esperamos que o sistema modifique seu modelo, refinando-o.

Outra dificuldade citada por (BURROWS; GUREVYCH; STEIN, 2015) está em encontrar os *datasets* utilizados por trabalhos da literatura, sendo que geralmente as informações coletadas na própria universidade não são disponibilizadas. Neste estudo esperamos realizar

comparações em, ao menos, três *datasets* que temos acesso e são conhecidos da literatura e dois *datasets* locais. Nesses *datasets* que temos disponíveis, apresentados em detalhes na Seção 6, encontramos uma variação no padrão de notas (contínuas ou discretas), na avaliação de um ou mais especialistas e na existência ou não de respostas candidatas. Tal variabilidade permite que diferentes testes sejam realizados para investigar o modelo avaliativo do professor, seja ele dado pela nota atribuída ou pela resposta candidata.

É importante descrever ainda a diferença entre o conjunto de informações selecionadas para a classificação correta das respostas e a interpretação avaliativa (RAMACHANDRAN; CHENG; FOLTZ, 2015). A separação do cunho interpretativo do interlocutor deve ser estritamente analisada conforme o modelo avaliativo. Se os padrões de nota e de resposta estiverem em conformidade com a avaliação do especialista, a redução de dimensionalidade dada pela otimização deve indicar uma boa interpretação da relação termo-classe. Assim, encaramos a avaliação como um processo constante e passível de revisões, para que o ajuste do sistema torne o modelo extraído cada vez mais próximos das expectativas do professor.

## 4 Objetivos

No projeto de pesquisa aqui descrito temos o intuito de ajustar o modelo de correção criado pela máquina aos padrões estabelecidos pelo professor através da sua avaliação por classe (grupo de nota). Segundo a consistência de cada grupo, reduziremos o esforço de correção do professor com a avaliação das respostas que apresentem apenas determinadas características textuais. Com padrões bem definidos, esperamos representar o critério avaliativo da questão justificando a classe atribuída através do seu respectivo sumário. Tal sumário, então, são os padrões de cada classe de nota partindo do agrupamento *a priori* das questões. É através desse sumário por nota que recuperamos um possível critério de correção. Desta forma, esperamos que o professor esteja apto para gerenciar o seu método avaliativo em um tempo menor enquanto concentra-se na verificação de aprendizagem do aluno.

Neste projeto, portanto, temos como objetivo aproximar o critério avaliativo do aluno com a definição de padrões de correção e a criação de *feedbacks*. Para isso, serão estudados

os padrões avaliativos do professor e os métodos de representação do conhecimento em base de dados de questões discursivas curtas. Para atingir o objetivo geral descrevemos os seguintes objetivos específicos:

- Estudar o impacto das técnicas de Processamento de Linguagem Natural e Recuperação da Informação para a identificação da relação termo-classe de forma léxica, morfológica, semântica, sintática, estatística ou espacial (BURROWS; GUREVYCH; STEIN, 2015; BUTCHER; JORDAN, 2010).
- Organizar *datasets* públicos e locais para comparação direta com resultados obtidos em estudos correlatos (BURROWS; GUREVYCH; STEIN, 2015).
- Unificar padrões de respostas dadas por professores e alunos, observando a frequência de ocorrência e co-ocorrência de termos segundo sua relevância (BUTCHER; JORDAN, 2010).
- Criar modelos avaliativos através do reconhecimento de padrões variáveis em categorias em dados discretos e contínuos (BURROWS; GUREVYCH; STEIN, 2015).
- Elaborar e ajustar modelos de acordo com a eficiência do sistema na recuperação da resposta atribuída pelo professor e seu modelo avaliativo (BURROWS; GUREVYCH; STEIN, 2015).
- Identificar a relação da avaliação com o comportamento textual da classe para remoção de *outliers* e manter a consistência da classificação (BUTCHER; JORDAN, 2010).
- Apresentar avaliações adequadas ao formato de correção do professor (BUTCHER; JORDAN, 2010).
- Gerar *feedbacks* que colaborem com o processo avaliativo, como o quadro de *rubrics*, de forma a contribuir com a discussão de resultados e a representação do critério de correção (OLIVEIRA et al., 2010).

## 5 Metodologia

O mapa de características foi uma técnica desenvolvida com base no método de avaliação de respostas discursivas (SPALENZA et al., 2016). Com esta ferramenta, o professor corrige cada questão com notas que serão interpretadas como classes no sistema para, automaticamente, identificar trechos relevantes para cada uma delas. Em geral o processo segue as etapas de coleta de dados, padronização, agrupamento, seleção de características e elaboração de relatórios, apresentados na Figura 1.

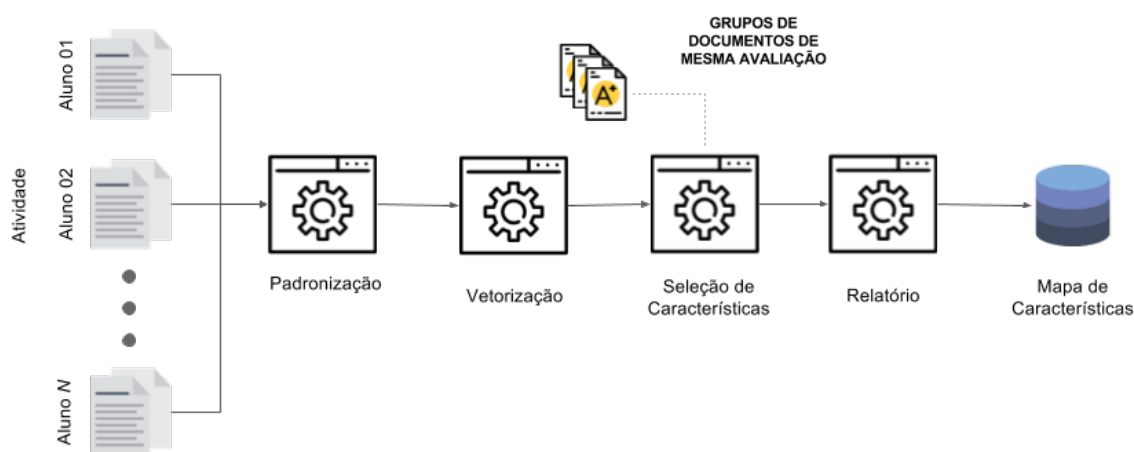


Figura 1: Etapas do mapa de características para captura do modelo avaliativo do professor.

Para a coleta de dados utilizamos o Ambiente Virtual de Aprendizagem - AVA Moodle<sup>2</sup>. O uso dessa plataforma é muito conhecido nos cursos EaD e conta também com várias aplicações de sucesso em turmas presenciais. A submissão de tarefas em texto *online* no AVA permite que a atividade, quando finalizada, seja coletada pelo *software* de extração de informações descrito por (PISSINATI, 2014) para ser processado pelo sistema.

Após a coleta o sistema realiza processos de padronização textual como a eliminação de sinais gráficos, *stemming* (radicalização), contagem de *n-grams* e remoção de *stopwords*. Essa etapa de padronização é uma tentativa de tratamento do conteúdo do documento para garantir equivalência entre as informações apresentadas. Após esse pré-processamento ocorre a vetorização de documentos para interpretação computacional do seu conteúdo. A vetorização adotada usa a frequência dos termos indexados a partir dos textos, ou *Term Frequency* -

<sup>2</sup>Modular Object Oriented Distance LEarning - Moodle - <moodle.org>

*TF* em inglês. Esse modelo se baseia na técnica de processamento de textos “*bag of words*” onde cada índice é associado a um termo distinto encontrado no conjunto de documentos.

No modelo vetorial, cada documento  $d$  de um conjunto  $D = \{d_1, d_2, d_3, \dots, d_{|D|}\}$  de documentos é representado como um vetor de termos  $t$ . Durante a vetorização, é verificada a frequência de ocorrência (TF) de cada um dos  $t$  termos em cada documento  $d$  em  $|D|$ , sendo  $|D|$  o total de documentos desse conjunto. Para cada  $d$ , portanto, é computada a frequência individual dos  $t = \{t_1, t_2, t_3, \dots, t_k\}$  termos que existem na base de dados, sendo  $k$  o número de termos distintos encontrados na coleção. Assim, cada documento  $d$  é representado como um vetor com as frequências individuais  $n$  de cada um dos  $k$  termos distintos, como o exemplo visto na Equação 1.

$$\begin{aligned} d_0 &= \{n_{t_1}, n_{t_2}, n_{t_3}, \dots, n_{t_k}\} \\ d_1 &= \{n_{t_1}, n_{t_2}, n_{t_3}, \dots, n_{t_k}\} \\ &\vdots \\ d_{|D|} &= \{n_{t_1}, n_{t_2}, n_{t_3}, \dots, n_{t_k}\} \end{aligned} \tag{1}$$

Depois do processo de vetorização ocorre a análise de similaridade dos documentos. Durante esse processo os documentos são clusterizados, particionados em treino e teste conforme os principais padrões selecionados nos grupos resultantes do algoritmo de *clustering*. Definimos o número de grupos *a priori* pelo agrupamento de melhor *silhouette score* (ROUSSEEUEW, 1987). Após a amostragem, os itens são avaliados pelo especialista e cada nota é considerada uma classe.

Para seleção das informações que representem a essência das tarefas, o mapa de características busca a resposta mínima de cada classe. Sabendo que o professor já efetuou a correção, o sistema agrupa as respostas com notas equivalentes. Em cada grupo o conjunto mínimo de resposta é recuperado com a análise iterativa da densidade do agrupamento com um Algoritmo Genético. Nesse método de otimização tentamos reduzir o número de termos selecionados avaliando o conjunto de informações da classe. Assim, a cada ciclo do algoritmo, o *fitness* é calculado pela razão entre o número de características selecionadas para esses documentos e a densidade de similaridade.

As características selecionadas são utilizadas para representar as partes significativas de cada grupo de documentos, resumizando-os. Com um peso atribuído pela frequência de ocorrência do termo na classe, uma visualização em HTML é gerada para discussão colaborativa dos resultados. Nessa visualização, exemplificada nas Figuras 2a e 2b, cada termo é colorido conforme a representatividade na sua classe correlata.


Através de um servidor WEB que irá armazenar os dados para futuras interações  
Criando campos ocultos que serão usados por scripts de processamento para armazenar informações  
As informações ficam armazenadas em um servidor web e através de um motor de busca buscador é possível que estas informações seja consultadas pelo usuário  
É possível obter informações de usuários em um documento eletrônico através de formulários a ser preenchido por ele um campo dentro do próprio documento  
De varias maneiras como formulários com metodos GET e POST Sessões e Cookies  
É possível devido ao banco de dados  
Basta usar um servidor para armazenar as informações enviadas pelo usuário e recuperá las mais tarde para uso  
É possível obter informações de leitor usuário através da criação de um formulário Existem duas partes no formulário a coleção de campo rótulos e botões e script de processamento  
Quando o usuário entra numa rede precisa preencher formulários que são salvos e enviados aos servidor sendo assim as preferências dele estarão registradas para os dados das próximas navegações  
Através da computação em nuvem  
Uma das praticas é o armazenamento da informação através dos dados em \_Formulários\_HTML como é feita a aplicação que receberá essas informações no \_lado\_ do servidor  
Por meio do formulário que possibilita a interação do usuário e do servidor é possível armazenar as preferencias dos usuários no banco de dados após a análise do Script  
É possível por meio de uma servidor onde o site esta hospedado que tem uma linguagem e uma folha de estilos e tenha ainda uma banco de dados  
O sistema operacional recebe os comandos enviados pelo usuário Os sistemas operacionais utilizam interfaces gráficas para facilitar a interação do usuário muitos comandos podem ser executados através dos  
icons  
Atraves de um serviços de Assinatura Digital de Documentos Eletrônicos  
Utilizando interação por formulário formulários estes ligados a bancos de dados  
Sim é possível utilizando tecnologias de construção de paginas na grande rede de computadores através de programação com html por exemplo  
Quando o usuário fornece suas informações em determinado lugar são gravadas e usadas como referência futuramente  
Por meio de formulários preenchidos pelo leitor usuário ao acessar tais documentos desde newsletter ou Comment Form até formulários de cadastro se for o caso  
repositórios de documentação científica podemos pensar na integração de repositórios digitais Conhecer as propriedades físicas dos suportes a serem utilizados para a migração Saber criar e manter metadados de preservação digital

#### LEGENDA

Nota 100.0 Nota 90.0 Nota 0.0

(a) Exemplo de visão geral em HTML para uma atividade.

### Feedback

Nota	100,00 / 100,00
Avaliado em	domingo, 12 Feb 2017, 11:08
Avaliado por	 Moodle WS
Comentários de feedback	<div> <div></div> <div>Exercício 5 Os elementos base no modelo proposto pelo DCC Digital Curation Center são Dados objetos digitais base de dados e também...</div> </div>

(b) Exemplo de visão do aluno com *feedback* no AVA.

Figura 2: Termos selecionados e ponderados com mapa de características apresentando a visão geral e individual para discussão dos resultados.

## 6 Bases de Dados e Experimentos Iniciais

Foram realizados três experimentos com o protótipo. O primeiro para verificar a adequação ao modelo de avaliação do especialista. Após isso, constatamos a relação de determinados termos por classe de nota em respostas discursivas curtas. Por fim, analisamos a relação do conteúdo do grupo de documentos com o método avaliativo do professor. Para realizar os testes temos disponíveis cinco bases de dados de questões discursivas:

- Ufes Disciplinas 2015-2016
- Ufes Vestibular 2012
- Curso de Introdução a Ciência da *Open University* da Inglaterra.
- Curso de Estrutura de Dados da Universidade do Norte do Texas (*Texas Dataset*)
- Competição ASAP da *Hewlett Foundation*

Testes prévios em quatro dessas bases de dados indicaram o potencial da ferramenta. O *dataset* da *Open University*, por ter sido recentemente adquirido, ainda não foi testado. Cada uma dessas bases de dados é descrita a seguir.

### 6.1 Base de Dados do Vestibular UFES

Conjunto de atividades discursivas transcritas do vestibular de 2012 da Universidade Federal do Espírito Santo - UFES da prova de língua portuguesa. No total são 460 respostas para as cinco questões avaliadas por dois avaliadores, contendo 92 cada. Na avaliação do vestibular, caso houvesse divergências de mais de 1 ponto entre as duas correções, um terceiro avaliador era acionado.



## 6.2 Base de Dados das Disciplinas UFES

Essa base de dados foi coletada de algumas disciplinas ministradas na Universidade Federal do Espírito Santo - UFES entre 2015 e 2016 através do Moodle do Laboratório Computação de Alto Desempenho - LCAD. Entre as disciplinas estão Metodologia e Técnicas de Pesquisa Científica, Filosofia e Tecnologia da Informação II. Totalizando 45 atividades, neste *dataset* foram recebidas 1162 submissões com média 25,82 e desvio padrão de 13,54 respostas por atividade.

O diferencial dessa base de dados é a mudança de características nas respostas encontradas nas múltiplas disciplinas, professores e alunos. Observam-se alterações consideráveis quanto ao tamanho, número de grupos, critério avaliativo ou objetividade.

## 6.3 Base de dados da Universidade do Norte do Texas (Inglês)

*Dataset* coletado com alunos de Estrutura de Dados do curso de Ciência da Computação na Universidade do Norte do Texas. Resultado do trabalho realizado por (MOHLER; BUNESCU; MIHALCEA, 2011), a base de dados, em sua segunda versão, constitui-se de dez atividades com até sete questões somados a dois exames com dez questões cada. As provas foram avaliadas em intervalos de 0 a 10 e as atividades de 0 a 5 pontos. A correção foi feita por dois professores para as respostas de, em média, 30 alunos participantes. Após a avaliação foi extraída a média entre os dois corretores. No total, essa base de dados contém 86 atividades e 630 respostas curtas e, para cada atividade, existe uma resposta criada pelo professor para correção.

## 6.4 Base de Dados do Concurso ASAG-SAS no Kaggle (Inglês)

A competição de avaliação automática de questões discursivas curtas - *ASAP - Automated Student Assessment Prize*, foi uma competição organizada no *Kaggle*. O *Kaggle* é uma plataforma de competições em Mineração de Dados que reúne tarefas complexas formadas por

problemas reais enfrentados por grandes empresas. A sua comunidade era de 536 mil usuários registrados em 2016, que disputam prêmios e compartilham aprendizados em análise estatística e ciência de dados.

No *Kaggle*, a organização *Hewlett Foundation* patrocinou três competições. As três foram denominadas como as seguintes fases da ASAP:

- Fase 1: Demonstração em respostas longas (redações);
- Fase 2: Demonstração em respostas curtas (discursivas);
- Fase 3: Demonstração simbólica matemática/lógica (gráficos e diagramas).

A Avaliação Automática de Respostas Curtas (*Automatic Short Answer Grader*) ASAG - SAS premiou com um total de 100 mil dólares os cinco primeiros colocados. A base de dados de respostas curtas, contém dez questões de artes a ciências. Ao final do concurso, os dados disponíveis avaliados por dois professores totalizam 17043 itens com aproximadamente 1700 itens por atividade.

## 6.5 Base de Dados da *UK Open University*

Utilizada por Butcher e Jordan (2010), essa base de dados foi coletada na disciplina de Introdução à Ciência da *Open University* da Inglaterra. Foram coletadas 20 questões que contém entre 511 e 1897 respostas cada, com em média 1247 respostas. Foram avaliadas por ao menos um especialista e pelo sistema *Intelligent Assessment Technologies FreeText Author*. A distribuição de classes é binária, marcando cada resposta como correta ou incorreta. Os diferenciais apresentados pelo *dataset* são a avaliação binária e o número de tentativas utilizadas pelo aluno.

## 6.6 Experimento: Conformidade com a Avaliação Humana

Esse experimento foi um comparativo com os resultados de Pissinati (2014), onde o autor relacionou a influência de diferentes tipos de pré-processamento na nota final. Os

testes foram realizados no *dataset* do vestibular da Universidade Federal do Espírito Santo de 2012. Para a construção da base de dados foram transcritas as 5 questões discursivas para a disciplina de Português, totalizando 460 respostas como foi descrito na Seção 6.1.

O melhor resultado alcançado por aquele autor nos testes foi de 1,36 pontos de erro médio absoluto e 1,54 pontos de desvio padrão do erro. Entre os dois especialistas avaliadores o erro alcançado foi de 1,46 pontos e com desvios de 1,57 pontos. No trabalho inicial, nossos testes alcançaram resultados de 1,25 pontos de erro médio absoluto e desvio padrão de 1,2 pontos.

Podemos inferir então que o sistema coletou adequadamente as principais respostas para treinamento e comparou-as adequadamente. Sabendo que Pissinati (2014) fez testes para adequação ao modelo avaliativo do professor através do pré-processamento, seus resultados se aproximaram do obtido entre humanos. Nosso modelo, sem testes direcionados para otimizar esses resultados, já alcançou 0,25 pontos de melhoria do valor entre humanos. Porém, os resultados daquele autor para um segundo avaliador foram regulares enquanto nosso sistema resultou em erros bem maiores. Isso indica a influência da análise linguística nos métodos de interpretação do conteúdo. Por isso, buscamos estudar de forma mais detalhada os padrões coletados e as técnicas de pré-processamento para padronização das ocorrências dos termos. Esperamos assim, estudar os resultados e propor melhorias na adequação com a avaliação do humano.

## 6.7 Experimento: Relação Termo-Nota

Esse experimento foi resultado de testes com dados coletados na universidade, que formaram o *dataset* de disciplinas de UFES (Seção 6.2) e da literatura (Seção 6.3). Nesses, buscamos identificar a relação do conteúdo objetivo por classe com a sumarização do conteúdo. Assim, para o estudo do modelo de classificação foi utilizado o Algoritmo Genético, como detalhado na Seção 5.

Na base de dados de disciplinas da UFES foram processadas 45 atividades com 1162 documentos. Na base da Universidade do Norte do Texas foram 86 atividades com 2415

documentos. A diferença na classificação após a seleção de características pelo Algoritmo Genético - GA foi testada com os classificadores *K-Nearest Neighbors Classifier* - *K-NN* e com o *Nearest Centroid Classifier* - *NC*. Os resultados são apresentados nas Tabelas 1 e 2.

Classificador	<i>Precision</i>		<i>Recall</i>	
	Antes do GA	Depois do GA	Antes do GA	Depois do GA
<i>K-NN</i>	89,52	89,93	81,54	81,70
<i>NC</i>	89,32	90,02	81,10	81,90

Tabela 1: Classificação do *dataset* da Universidade Federal do Espírito Santo antes e depois da seleção de características.

Classificador	<i>Precision</i>		<i>Recall</i>	
	Antes do GA	Depois do GA	Antes do GA	Depois do GA
<i>K-NN</i>	88,48	87,23	80,98	78,29
<i>NC</i>	90,57	84,72	84,46	72,20

Tabela 2: Classificação do *dataset* da Universidade do Norte do Texas antes e depois da seleção de características.

Podemos ver pelas Tabelas 1 e 2 que o sistema teve desempenhos equivalentes para a base de dados da UFES e *Texas Dataset*. Essa consideração é válida apesar da ocorrência de uma pequena melhoria em *precision* e *recall* na primeira e uma pequena queda na segunda. Porém, a quantidade de informações para classificação após o GA foi de, respectivamente, 24,84 % e 11,48% das características. Assim, esperamos que o estudo aprofundado da seleção termo-nota permita que com pouca informação sejam realizadas classificações mais equivalentes ao modelo do especialista. Ainda através da construção desses sumários de resposta, após a classificação, esperamos que o sistema seja capaz de representar seu modelo avaliativo pelo mapa de características. Essa queda na classificação apresentada na Tabela 2,

ainda foi alvo de um estudo sobre a extração do critério avaliativo apresentado na próxima seção.

## 6.8 Experimento: Conteúdo-Chave para Avaliação

Em geral o mapa de características tem um efeito positivo nas bases de dados processadas. A redução de dimensionalidade quando existe uma tendência de resposta condiciona os classificadores a interpretarem melhor o critério de correção do professor. Aprendendo o modelo do professor, por consequência, torna mais simples relatar a forma avaliativa da máquina e a discussão dos resultados. Podemos ver isso, por exemplo, na atividade 1.5 apresentada na Figura 3.

Para a atividade 1.5 os alunos responderam a seguinte pergunta: “*what is a variable?*”. Nas respostas vemos marcações que relacionam quatro notas distintas com os termos. Essas marcações variam em 10 tons de coloração de 0, equivalente a uma baixa correlação, até 9 com alta correlação à nota. Os termos não marcados são dados como de baixa relevância pelo sistema. As alterações de tonalidade são dispostas em até 6 cores vinculadas cada qual a uma classe de nota.

Na Figura 3 podemos ver que as palavras *location*, *memory* e *value* são fortemente ligadas com a nota máxima 5,0 em azul. Enquanto isso *program* e *string* pertencem ao grupo de nota 4,0 em amarelo, *variable name* e *stored* ao de nota 3,0 em verde e *data* e *assigned* remetem à nota 2,0 em vermelho. Palavras pouco correlacionadas como *symbol*, *example*, *computers* e *programmer* são consideradas neutras. Porém, se compararmos as marcações com a resposta esperada apresentada pelo professor “*a location in memory that can store a value*”, temos as principais palavras diretamente relacionadas ao critério de correção.

Porém, como visto na classificação dada na Seção 6.7, a perda de informação durante a redução pode afetar negativamente a ação do classificador. No *Texas dataset*, a queda de 5,85% apresentada na Tabela 2 representa esses casos. Na atividade 2-6, por exemplo, ocorrem avaliações que não caracterizam termos correlatos com a nota atribuída. Isso ocorre para a pergunta “*what is the difference between a function prototype and a function definition?*”. Os documentos de resposta são apresentados na Figura 4.

**Variable** can be a integer or a **string** in a **program**

In **programming** a **structure** that **holds data** and is **uniquely named** by the **programmer** It **holds** the **data assigned** to it until a **new value** is **assigned** or the **program** is **finished** br

It is a **location** in the **computer s memory** where it can be **stored** for **use** by a **program**

A **location** in **memory** where **value** can be **stored**

a **value word** that can assume any of a set of **values**

A pointer to a **location** in **memory**

A **variable** is the **memory** address for a specific type of **stored data** or from a mathematical perspective a **symbol** representing a fixed definition with **changing values**

A **variable** is a **value** that is subject to **change** in a **computer s memory** that can be **used** by **programs** **Programs** can **change** the **value** of the **variable** and recall it later or act on it directly

a **symbol** that **stands** in for a **value** that **may** or **may not change** depending on the **program**

a placeholder to **hold** information **used** in the **program** br for **example** br **int** can **hold** 1 2 3 4 68 72 256 etc br **float** can **hold** 1 54 55 55 1 24 5 657 8 8123 et br **char** can **hold** A B C D E F 4 5 6 P etc br

a **stored value** used by the **program**

A way to **store** different **values** into the **program** such as **numbers words** letters etc

A **variable** is a **location** in **memory** where a **value** can be **stored**

An object with a **location** in **memory** where **value** can be **stored** br

**location** in **memory** where a **value** can be **stored**

a **variable** is an object where **data** is **stored**

**Location** in **memory** where a **value** can be **stored**

it is a **location** in **memory** where **value** can be **stored** br

A **variable** is the **location** in a **computer s memory** where a **value** can be **stored** for **use** by a **program**

A **variable** is a **location** in a **computers memory** where a **value** can be **stored** br br for **use** by a **program**

a **location** in **memory** where **data** can be **stored** and retrieved

Is a method or identifier I **would say** we **use** to bind a **data** object to **memory location** which is then **stored** in a **location** that can be accessed when and manipulated later when the **variable name** is called

**Variable** is a **location** in the **computer s memory** in which a **value** can be **stored** and later can retrieve that **value**

A **variable** is a **location** in the **computer s memory** where a **value** can be **stored** for **use** by a **program** Each **variable** has a **name** a **value** a type and a **size**

A **named** object that can **hold** a numerical or letter **value**

It s a sybol or **name** for a **value number** **Example** a **\_used\_number** can **stand** for any given **number** and the **programmer** can refer to that **number** by **using** the **variable name**

A **variable** is a **location** in the **computers memory** where a **value** can be **stored** for **use** by a **program**

An identifier that **holds** a **location** in **memory**

a block of **memory** that **holds** a specific type of **data**

## LEGENDA

**Nota 5.0** **Nota 4.0** **Nota 3.0** **Nota 2.0**

Figura 3: Mapa de características de todos os alunos para a atividade *DS CC UNT 1-5*.

Podemos analisar a Figura 4 paralelamente com a chave de correção do professor para verificar os possíveis motivos da quantidade de marcações neutras. A resposta aguardada era “*a function prototype includes the function signature, i.e., the name of the function, the return type, and the parameters’ type. The function definition includes the actual body of the function*”. Pelos documentos da figura e a chave de correção vemos que a enumeração de partes de uma função era a essência da resposta. Porém, ao contrário do aguardado, não temos intercessão observável exceto em *function prototype* e *function definition*. Tais trechos

A **function definition** does not **require** any **additional information** that **needs** to be **passed inside** its **parenthesis** **br** to **execute** While a **definition** **prototype** **requires** more than **one parameters** to be **passed in order** to **complete** its **br** **task**

The **FUNCTION PROTOTYPE** is where the **programmer declares** that he she is **using** a **function** other than **main** This is **like declaring** a **variable** the **programmer knows** that he she will be **using** in the **future** but has **yet** to **say** where they are **going** to **use** it or how This is **answers** the **question** who it **gives** the **function** a **name** and **character** The **function prototype** by **common practice** is **placed** at the **beginning** of the **program** after the **includes** and before **main** **br** The **FUNCTION DEFINITION** is the **guts** of the **function** This is where the **programmer decides** what the **function** is **going** to do and **tells** it how to do it It **takes whatever information** it is **given** and **performs** the **operations** It works sort of like the **brain** the **brain** takes in **input** and **based upon** that **input** **performs** in some way **producing** an **output** The **function definition** is **placed outside** of **main** and any other **functions** A **function** is its own **entity** and should be **thought** of as such

**Function definitions** are just that the **definition** The **prototype** is what the **compiler** uses to **check** that **calls** to **function** are **correct**

a **prototype** does not **include** any **actual code** where the **function** has all the **code** that is **executed** in the **program**

a **prototype** **declares** what will be **used** in the **program** and the **definition**

A **function prototype** **lays out** the **name** **return type** and the **number** and **types** of **parameters** the **function** **expects** to **receive** in a **certain order** The **details** for **function prototypes** are in the **function definition**

A **function definition** is the **code** that **defines** the **function** **placed** in the **brackets** that **determines** that **function** s **operation** **br** **br** A **function prototype** shows the **function** s **public interface** without **exposing** **implementation** It shows **name** **return type** and **type** of **paramaters**

A **function prototype** is just a **declaration** of the **function** **existing** and **cant** be **used** as an **actual function** A **function** has to be **created** with a **definition** **within** to **tell** the **compiler** what the **function** does

the **Prototype** **creates** a **framework** to **call** the **function** **definition** While a **function definition** is where the **function** is **actually** **programmed** out and **created** into a **final product**

A **function prototype** **describes** the **class** s **public interface** without **providing** how the **function** works **br** A **function definition** contains the **inner workings** of the **function**

**prototype** **states** all **functions** in that **class** before **compilation** where the **definition** **actually** **holds** the **source** for the **functions**

**Function prototypes** **tell** the **compiler** the **function** **names** its **return type** and the **types** of its **parameters** where as **function definitions** **actually** **implement** the **member functions**

A **function prototype** **tells** the **compiler** the **function** **name** **return type** and the **number** and **type** of **parameters** without **revealing** the **implementations** contained in the **function definition**

A **function prototype** just **specifies** **parameters** A **function definition** **includes** **parameters** and a **code body**

**Function prototypes** **describe** the **class** s **public interface**

A **prototype** only **declares** **name** **return type** and **input type** **br** A **definition** **also** **defines** the **scope** **variables** **process** and **return function**

a **function prototype** simply **declares** the **functions** **parameters** the **function definition** **includes** any **necessary variables** and the **function** s **actual code**

A **function prototype** is a **declaration** of a **function** while **function definition** **specifies** what a **function** does

in a **function prototype** you **include** the **return type** the **name** of the **function** and its **parameters** if any are **needed** **br** **br** in a **function definition** you **write** the **code** of what the **function** will do

A **function definition** contains all the **code** for a **function** to **work** A **function prototype** just **shows** the **output** **input** and **function name**

**function prototype** **describe** the **class** s **public interface** without **revealing** **br** **br** the **class** s **member function** **implementations** **function definitions** **show** what **br** **br** **implementations** are being **done**

A **function prototype** only **names** the **function** its **return type** and it s **argument list** while a **definition** **defines** the above as **well** as what the **function** **actually** does

**function prototype** **describes** a **classes** **interface** without **reviling** whatever is **inside** as for the **function definition** **can** t do that **br**

**Function prototype** is a **declaration** **Function definitions** w **multiple parameters** often **require** more than **one piece** of **information** to **perform** their **tasks**

A **function prototype** is a **declaration** of a **function** that **tells** the **compiler** the **function** s **name** its **return type** and the **types** of its **parameters**

A **prototype** **shows** only **return types** and **necessary parameters** The **definition** **includes** **names** for those **parameters** and **defines** what the **object** is **actually capable** of doing

**prototype** only **tells** the **user** what **data** **types** **go** into a **function** and what **type** is **returned**

**Function prototype** is **located** in the **h file** and only contains the **access** **function name** and **paramater type** **br** **br** **Function definition** contains the **code** for the **function** to **perform** its **activity**

a **function prototype** is **used** to **reference** the **compiler** to a **function** that will be **defined** later on a **function definition** is the **actual** **function** itself **complete** with **return type** **parameters** etc

A **function prototype** **describes** a **class** s **public interface** without **revealing** the **class** s **member function** **implementations**

---

#### LEGENDA

Nota 5.0 Nota 4.0 Nota 1.0

Figura 4: Mapa de características de todos os alunos para a atividade *DS CC UNT 2-6*.

são encontrados nas respostas das três notas disponíveis (5,0 3,0 e 1,0). Assim, o processo de avaliação foi criterioso quanto a diferenciação entre as funções, independentemente do aspecto elencado. Isso confunde o algoritmo pela ausência de termos semelhantes por classes de nota ao coexistirem várias respostas possíveis.

O intuito deste projeto, portanto, é investigar com mais detalhes a atribuição de nota. Sabendo que existe, em geral, um modelo avaliativo do professor, esperamos reconstituir seu padrão para contribuir com o método avaliativo. Assim, ao relacionar termos com notas específicas esperamos criar modelos mais próximos do professor para a máquina, de forma mais adequada, colaborar com o processo de aprendizado. Com conhecimento adquirido sobre a disciplina através da avaliação, podemos atribuir funções mais específicas ao sistema, apoiando melhor a correção e gerando feedbacks mais próximos da resposta esperada.

## 7 Cronograma

A Tabela 3 apresenta o planejamento das atividades a serem realizadas durante o doutorado, inclusive os pré-requisitos previstos no edital do Programa de Pós Graduação em Informática da Universidade Federal do Espírito Santo - UFES. São descritas as 15 atividades previstas com os respectivos semestres que serão trabalhadas.

Para elaboração de artigos científicos visamos abordar a problemática citada nesse projeto na Seção 3. A perspectiva de publicações é dada em 4 temas principais. Inicialmente o tema a ser trabalhado é a definição de critério avaliativo e sua aplicação no método semi-automático. Posteriormente, serão analisados os métodos de ponderação por características textuais (termos ou *n-grams*) para visualização da informação e avaliação, aplicando o mapa de características como *feedback* aos alunos. O terceiro tema trabalhado será o impacto de modelos de agrupamento e remoção de *outliers* textuais na identificação de grupos de nota. Por fim, no quarto tema, serão estudadas as marcações aplicadas para recomendação de aprendizado.

Das publicações previstas, os temas têm estreita relação com os artigos publicados por Ramachandran, Cheng e Foltz (2015), Ramachandran e Foltz (2015), Butcher e Jordan



Id.	Descrição	Semestre	
		Início	Fim
1	Estudo da literatura para reconhecimento de padrões textuais	1	1
2	Estudo dos métodos da literatura para modelagem do avaliador	1	1
3	Organização dos <i>datasets</i> acessíveis	2	2
4	Estágio Docência	2	4
5	Disciplinas	2	4
6	Adequação do protótipo inicial	2	4
7	Implementação de modelos de recuperação de informação	3	5
8	Implementação de modelos de avaliação	4	6
9	Implementação de modelos de <i>feedback</i>	5	6
10	Artigos Científicos	3	6
11	Qualificação	5	5
12	Preparação para defesa	6	7
13	Exame de Proposta de Tese	6	7
14	Elaboração da Tese	7	8
15	Defesa da Tese	8	8

Tabela 3: Etapas previstas para alcançar o número mínimo de créditos exigido pelo programa.

(2010), Mohler, Bunescu e Mihalcea (2011), Burrows, Gurevych e Stein (2015) e Oliveira, Ciarelli e Oliveira (2013). As duas primeiras publicações foram publicadas no *Workshop on Innovative Use of NLP for Building Educational Applications* da *Conference on Empirical Methods in Natural Language Processing - EMNLP*. A 3ª foi publicada no *journal Computers & Education*. A 4ª foi para o *Annual Meeting of the Association for Computational Linguistics - ACL*. A 5ª, principal revisão de literatura encontrada da área, foi publicada no *International Journal of Artificial Intelligence in Education*. E a 6ª foi publicada no *journal Expert Systems with Applications*. O trabalho inicial deste projeto de pesquisa (SPALENZA et al., 2016), foi publicada durante o período do mestrado no Simpósio Brasileiro de Informática na Educação. Todas as publicações acima são atualmente qualificadas de acordo com os pré-requisitos previstos no edital do PPGI. Além disso, todos os sete trabalhos foram

pesquisas relacionadas ao processo de avaliação automática e que citam dados que temos acesso, sendo passíveis de serem reproduzidos em nosso sistema.

## Referências

- ARTER, J. A.; CHAPPUIS, J. *Creating & Recognizing Quality Rubrics*. Nee York, USA: Pearson Education, 2006. (Assessment Training Institute, Inc Series).
- BAILEY, S.; MEURERS, D. Diagnosing Meaning Errors in Short Answers to Reading Comprehension Questions. In: *Proceedings of the 3rd Workshop on Innovative Use of NLP for Building Educational Applications, held at ACL 2008*. Columbus, Ohio, USA: Association for Computational Linguistics, 2008. p. 107–115.
- BARREIRA, C.; BOAVIDA, J.; ARAÚJO, N. Avaliação Formativa: Novas Formas de Ensinar e Aprender. *Revista Portuguesa de Pedagogia*, Universidade de Coimbra, v. 40, n. 3, p. 95–133, 2006.
- BURROWS, S.; GUREVYCH, I.; STEIN, B. The Eras and Trends of Automatic Short Answer Grading. *International Journal of Artificial Intelligence in Education*, v. 25, n. 1, p. 60–117, 2015.
- BUTCHER, P. G.; JORDAN, S. E. A Comparison of Human and Computer Marking of Short Free-Text Student Responses. *Computers & Education*, v. 55, n. 2, p. 489 – 499, 2010.
- GOMAA, W. H.; FAHMY, A. A. Short Answer Grading Using String Similarity And Corpus-Based Similarity. *International Journal of Advanced Computer Science and Applications(IJACSA)*, v. 3, n. 11, 2012.
- LEFFA, V. J. Análise Automática da Resposta do Aluno em Ambiente Virtual. *Revista Brasileira de Linguística Aplicada*, SciELO, v. 3, p. 25 – 40, 00 2003.
- MOHLER, M.; BUNESCU, R.; MIHALCEA, R. Learning to Grade Short Answer Questions Using Semantic Similarity Measures and Dependency Graph Alignments. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (HLT '11), p. 752–762.
- NOORBEHBAHANI, F.; KARDAN, A. A. The automatic assessment of free text answers using a modified bleu algorithm. *Computers & Education*, Elsevier Science Ltd., Oxford, UK, UK, v. 56, n. 2, p. 337–345, feb 2011.
- OLIVEIRA, E. et al. Combining Clustering and Classification Approaches for Reducing the Effort of Automatic Tweets Classification. In: *KDIR 2014 - Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*. Rome, Italy: ScitePress, 2014. v. 6, p. 465–472.

- OLIVEIRA, E. et al. Uma Tecnologia de Agrupamento de Respostas para Redução de Esforço de Correção de Atividades em Sistema Online de Apoio à Avaliação Formativa em Indexação. In: *XI Encontro Nacional de Pesquisa em Ciência da Informação*. Rio de Janeiro, RJ, Brasil: Associação Nacional de Pesquisa e Pós-Graduação em Ciência da Informação (Ancib), 2010.
- OLIVEIRA, K. L. d.; SANTOS, A. A. A. Compreensão em Leitura e Avaliação da Aprendizagem em Universitários. *Psicologia: Reflexão e Crítica*, SciELO, v. 18, p. 118 – 124, 04 2005.
- OLIVEIRA, M. G.; CIARELLI, P. M.; OLIVEIRA, E. Recommendation of Programming Activities by Multi-label Classification for a Formative Assessment of Students. *Expert Syst. Appl.*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 40, n. 16, p. 6641–6651, nov 2013.
- PAIVA, R. et al. Mineração de Dados e a Gestão Inteligente da Aprendizagem: Desafios e Direcionamentos. In: *I Workshop de Desafios da Computação Aplicada à Educação (DesafIE!2012)*. Curitiba, PR, Brasil: Sociedade Brasileira de Computação, 2012. v. 1.
- PÉREZ-MARÍN, D.; PASCUAL-NIETO, I.; RODRÍGUEZ, P. Computer-Assisted Assessment of Free-Text Answers. *The Knowledge Engineering Review*, Cambridge University Press, v. 24, n. 4, p. 353–374, 2009.
- PISSINATI, E. M. *Uma Proposta de Correção Semi-Automática de Questões Discursivas e de Visualização de Atividades para Apoio à Atuação do Docente*. Dissertação (Mestrado) — PPGI - Universidade Federal do Espírito Santo, Vitória, ES, Setembro 2014.
- RAMACHANDRAN, L.; CHENG, J.; FOLTZ, P. W. Identifying patterns for short answer scoring using graph-based lexico-semantic text matching. In: *BEA@NAACL-HLT*. Denver, Colorado: Association for Computational Linguistics, 2015.
- RAMACHANDRAN, L.; FOLTZ, P. W. Generating reference texts for short answer scoring using graph-based summarization. In: *BEA@NAACL-HLT*. Denver, Colorado: Association for Computational Linguistics, 2015.
- ROUSSEUW, P. J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, v. 20, p. 53 – 65, 1987.
- SPALENZA, M. A. et al. Uso de Mapa de Características na Avaliação de Textos Curtos nos Ambientes Virtuais de Aprendizagem Classes de Respostas Discursivas. In: *Simpósio Brasileiro de Informática na Educação (SBIE)*. Uberlândia, MG, Brasil: Sociedade Brasileira de Computação, 2016. v. 27.