Detecção de Similaridade entre Modelos Entidade Relacionamento para fins educacionais

Pedro Leonel T. Fernandes[1], **Sergio L. S. Mergen**[2]

[1]Centro de Tecnologia – Universidade Federal de Santa Maria (UFSM)
97105-900 -– Santa Maria -– RS — Brasil

[2]Departamento de Linguagens e Sistemas de Computação
Universidade Federal de Santa Maria (UFSM)

{plfernandes,mergen}@inf.ufsm.br

Abstract. *This paper proposes a similarity detector between conceptual models of the Entity-Relationship (ER) type. The detection approach is optimistic, which means it seeks to maximize similarity between two entities when there is a chance that they correspond to each other. The experiments show how this approach can be useful for educational purposes, whether for detecting plagiarism or for correcting modeling tasks.*

Resumo. *Esse artigo propõe um detector de similaridade entre modelos conceituais do tipo Entidade-Relacionamento (ER). A abordagem de detecção é otimista, ou seja, ela busca maximizar a similaridade entre duas entidades quando for identificada a possibilidade de que elas possuam alguma correspondência. Os experimentos mostram como essa abordagem pode ser útil em ambientes educacionais, seja para a detecção de plágio ou para a correção de tarefas de modelagem.*

1. Introdução

Dentre as principais técnicas para modelagem conceitual, a mais difundida foi a apresentada em 1976 por Peter Chen, o Modelo Entidade-Relacionamento (ER). O modelo é largamente utilizado para concepção inicial de um banco de dados ou até mesmo documentação de um banco de dados existente.

A principal causa da popularidade deste modelo está concentrada em sua fácil utilização e legibilidade, tanto por parte dos projetistas de sistemas com conhecimento na área quanto por parte dos usuários finais desses sistemas, que participam das etapas iniciais do levantamento de requisitos. Devido a sua simplicidade, o modelo também é largamente utilizado no meio acadêmico como método de aprendizagem nas disciplinas referentes a banco de dados.

Com o crescente uso de computadores pessoais para a realização de tarefas acadêmicas à distância (motivada em parte pela covid-19), surge a possibilidade de controlar de maneira mais efetiva os trabalhos entregues. Nas disciplinas voltadas ao ensino de bancos de dados, caso os trabalhos envolvem o desenvolvimento de modelos conceituais, o controle pode ser alcançado através de métodos computacionais que identifiquem similaridades entre dois modelos. Entre as benefícios oriundos da detecção de similaridades pode-se destacar a detecção de plágio (comparando tarefas entregues entre si) e a

verificação da correção (comparando uma tarefa entregue com um modelo usado como gabarito).

Neste contexto, este artigo propõe um método otimista que identifica o grau de similaridade entre dois modelos conceituais. O resultado é exibido como uma lista contendo, para cada entidade de um modelo, a entidade mais similar do outro modelo. Além disso, é retornado um escore normalizado que indique a similaridade global entre os dois modelos comparados.

Este trabalho está organizado da seguinte forma: A seção 2 apresenta os trabalhos relacionais. A seção 3 apresenta o processo de detecção de similaridade proposto. A seção 4 apresenta os experimentos. Por fim, a seção 5 destaca as considerações finais.

2. Trabalhos Relacionados

Há poucos trabalhos na literatura que sejam especificamente voltados ao uso de modelos ER em ambientes educacionais. Em suma, foi encontrado o interesse em desenvolver linguagens específicas de domínio para a modelagem conceitual, com o intuito de auxiliar no ensino de modelagem no meio acadêmico [Dimitrieski et al. 2015].

Com relação à detecção de similaridade entre modelos distintos, os trabalhos encontrados focam em estruturas de dados mais específicas, como documentos XML [Algergawy et al. 2010], ou mais enriquecidas, como ontologias [Shvaiko and Euzenat 2011]. No caso dos documentos XML, o trabalho de [Tahraoui et al. 2013] destaca que a principal preocupação é em detectar semelhanças morfológicas (estruturais), sem se atentar às semelhanças linguísticas. No caso das ontologias, além da preocupação em encontrar correspondências linguísticas e estruturais, alguns trabalhos também se preocupam com a correspondência semântica [Jean-Mary et al. 2009].

De modo geral, independente do tipo de modelo estudado, existem inúmeros trabalhos explorando os aspectos linguísticos, estruturais e semânticos. Boa parte deles explora aspectos que são encontrados em modelos conceituais. No entanto, o objetivo destes trabalhos é encontrar correspondências como forma de facilitar a integração de dados. O uso da similaridade para facilitar a avaliação de tarefas de modelagem exige abordagens diferentes, como será visto ao longo do artigo.

3. Proposta

O objetivo principal deste trabalho é elaborar uma ferramenta que exibe o quão similares dois modelos conceituais são entre si, para fins de avaliação de tarefas de modelagem. Para atingir esse objetivo, propõe-se um método composto por uma série de etapas, conforme ilustrado na Figura 1.

As etapas 1 e 2 correspondem à criação e representação dos modelos conceituais. A partir da etapa 3, cada entidade E_i de um modelo será comparada com cada entidade E_j do segundo modelo, o que gerará um escore de similaridade. Os escores gerados por cada etapa serão agregados na etapa 6, e utilizados para inferir um valor de similaridade geral entre os dos modelos. As etapas serão apresentadas em detalhes mais adiante.

Convém salientar os seguintes aspectos:

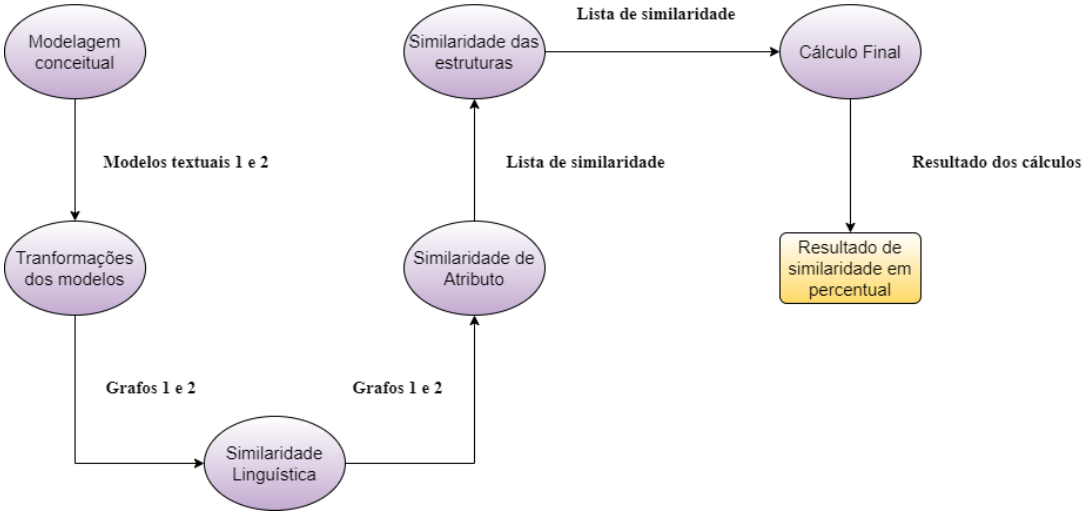

Figura 1. Etapas da Detecção de Similaridade

- A abordagem de casamento é otimista, ou seja, busca maximizar o valor de similaridade entre as entidades dos dois modelos.
- A abordagem de casamento busca detectar a presença do modelo 1 dentro do modelo 2.

Essas escolhas foram tomadas por dois motivos: Em tarefas de correção frente a uma gabarito, a abordagem adotada valoriza respostas que sejam variações da versão considerada correta, ou que resolvam parte do problema maior (considerando o modelo 2 como sendo o gabarito). Já em tarefas de verificação de plágio, essa abordagem tem maior potencial de detectar similaridades. Para evitar injustiças produzidas por falsos-positivos, cabe ao usuário verificar os casos detectados para confirmar a ocorrência de plágio. A seção 4 apresenta experimentos que exploram o potencial desta abordagem.

3.1. Etapa 1: Modelagem Conceitual

Esta etapa visa permitir que o usuário crie modelos conceituais através de uma linguagem de domínio específico para a modelagem conceitual. A linguagem proposta permite representar entidades, atributos, relacionamentos e atributos identificadores.

A Figura 2 apresenta um modelo conceitual de venda de produtos sendo descrito a partir da ferramenta. As palavras reservadas 'entity', 'relation' e 'attribute' são usadas para representar entidades, relacionamentos e atributos, respectivamente. Nos três casos, são usadas vírgulas para separar os elementos que compõem cada tipo de objeto.

Para objetos do tipo entidade, o primeiro elemento corresponde ao nome. Os demais correspondem aos atributos. A palavra reservada 'key' indica que o atributo consequente faz parte do identificador da entidade. No caso dos relacionamentos, o primeiro elemento corresponde ao nome do relacionamento. Após o nome, dois elementos devem ser informados, indicando os nomes das entidades que participam do relacionamento. Também pode-se usar 'key' para indicar que a entidade consequente faz parte do identificador do relacionamento. Para cada entidade, devem ser especificadas as cardinalidades

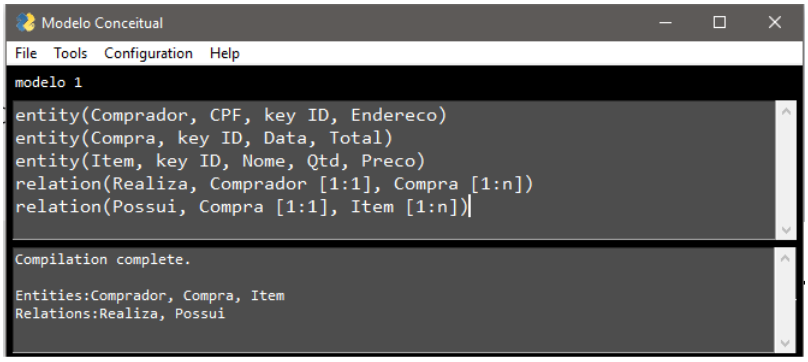

Figura 2. Ferramenta Criada para a Definição do Modelo ER

mínima min e máxima max usando a notação $[min, max]$ e considerando $\{0, 1\}$ e $\{1, n\}$ como valores possíveis para as cardinalidades mínima e máxima, respectivamente.

Por fim, o construtor 'attribute'é usado para indicar os atributos que pertencem a objetos do tipo relacionamento. Esse construtor também pode ser usado como forma opcional de especificar os atributos de entidades. O primeiro elemento de um 'attribute' é o nome do objeto para o qual serão informados os atributos. Os demais elementos indicam atributos que pertencem ao objeto informado. A palavra reservada 'key' também pode ser usada aqui caso algum dos atributos tenha função de identificação.

No exemplo descrito na Figura 2, tem-se que um item de nota está associado a uma compra, que por sua vez está associada a um comprador. Por outro lado, um comprador pode realizar várias compras, e em uma compra podem aparecer vários itens.

O código referente ao modelo conceitual é submetido a um parser que avalia os aspectos léxicos e sintáticos. Caso o modelo tenha sido descrito da forma correta, é exibida uma representação visual em formato UML. Caso contrário, é exibida uma mensagem que remete o usuário ao erro cometido. A ferramenta possui algumas limitações. Em primeiro lugar, não há suporte para recursos avançados, como especialização, relacionamentos n-ários e entidades associativas. Como esses construtores são pouco usados frente aos demais, sua implementação foi deixada para trabalhos futuros.

3.2. Etapa 2: Transformações dos modelos

Para realizar o processo de detecção de similaridade, os modelos criados na etapa de modelagem devem ser carregados e transformados em uma representação em memória, na forma de um grafo dirigido e valorado.

Um exemplo é ilustrado na Figura 3, que exibe o grafo referente ao modelo relacional apresentado na Figura 2. Neste grafo, as entidades são os vértices, enquanto os relacionamentos são as arestas, cujos valores armazenam as respectivas cardinalidades. Os atributos são informações que pertencem ao vértice, e não foram representados na figura para fins de legibilidade.

3.3. Etapa 3: Similaridade linguística

A similaridade linguística (S_n) calcula um valor de similaridade entre duas entidades com base nos seus nomes. A similaridade é derivada a partir de uma função chamada sim, que recebe como argumento os nomes das entidades. Dadas duas strings S_i e S_j, a função

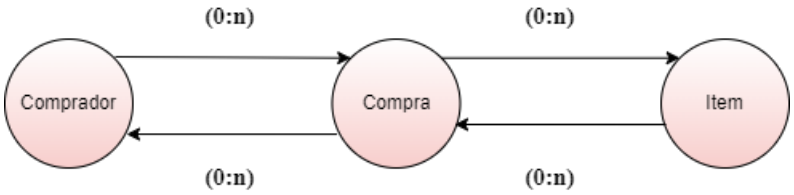

Figura 3. Grafo Representativo do Modelo Conceitual

$sim(s_i, s_j)$ encontra um escore de similaridade normalizado entre zero e um, sendo que, quanto maior o valor, mais alta é a similaridade. A Equação 1 apresenta como este valor é encontrado.

$$sim(S_i, S_j) = max(lcs(S_i, S_j), sinon(S_i, S_j)) \tag{1}$$

Conforme apresentado, a função sim retorna o maior valor dentre as funções lcs e $sinon$, cujas definições estão especificadas a seguir:

Longest Common Subsequence(lcs): Este algoritmo retorna o número de caracteres que pertençam à maior subsequência (cadeia de caracteres não necessariamente consecutivos) entre duas strings [Bergroth et al. 2000]. Dadas duas strings S_1 e S_2, e considerando que $N(S_1, S_2)$ seja a função que retorna o número de caracteres comuns entre S_1 e S_2, usa-se a Equação 2 para obter um valor de similaridade normalizado:

$$lcs(S_i, S_j) = \frac{N}{min(|S_i|, |S_j|)} \tag{2}$$

O uso do menor tamanho entre S1 e S2 como fator de normalização ($min(|S_i|, |S_j|$) é importante para detectar abreviaturas (quando os caracteres de uma abreviatura existem dentro de uma sequência não contígua dentro da string maior) e substrings (quando uma string está contida dentro de outra).

Correspondência de sinônimo ($sinon$): Esta função identifica se duas strings possuem alguma relação de sinonímia. Dadas duas strings, usa-se uma API para retornar uma lista de sinônimos para cada string. Essas duas listas são comparadas entre si. Caso haja alguma correspondência, o valor retornado é '1'. Caso contrário, o retorno é '0'. Para a língua portuguesa, foi usada a API *pysinonimos*, disponível em Python [1]. Essa API acessa o site brasileiro 'sinônimos' para efetuar sua busca. Já para a língua inglesa foi usada a API do Wordnet.

Para exemplificar, considere que o objetivo seja identificar as similaridades entre os dois modelos conceituais ilustrados na Figura 4. Tratam-se de modelos cujas entidades representam a mesma realidade, porém, com definições diferentes, seja no tocante aos atributos definidos ou aos próprios nomes escolhidos para as entidades.

A Tabela 1 apresenta os valores de similaridade linguística S_n encontrados para cada par de entidades. Os valores destacados em negrito representam, para cada entidade do modelo 1, qual entidade do modelo 2 obteve o maior escore.

[1] `https://github.com/diegofsousa/pysinonimos`

entidade 1	entidade 2	S_n	S_a	S_e	escore
comprador	**cliente**	0.70	0.33	**0.74**	**0.70**
	nota_fiscal	0.22	0.37	**0.78**	0.60
	prod_vendido	0.44	**0.72**	0.54	0.64
compra	cliente	0.17	0.35	**0.42**	0.37
	nota_fiscal	0.33	0.68	**0.84**	**0.74**
	prod_vendido	0.33	**0.72**	0.42	0.59
item	cliente	**0.75**	0.56	0.54	0.67
	nota_fiscal	0.25	0.43	**0.58**	0.5
	prod_vendido	0.25	**0.90**	0.74	**0.79**
Escore Final					**0.74**

Tabela 1. Similaridade Calculada entre Entidades

3.4. Etapa 4: Similaridade de Atributo

O cálculo da similaridade de atributos usa a função sim (detalhada na seção 3.3) para encontrar escores de similaridade entre atributos das duas entidades comparadas. Considerando que $E = \{a_1, ..., a_n\}$ refira-se à lista de nomes de atributos a_j que pertencem à entidade E, o valor é calculado com base na Equação 3.

$$S_a(E_i, E_j) = \frac{\sum_{a_x \in E_i} \arg\max_{a_y \in A(E_j)} sim(a_x, a_y)}{|E_i|} \tag{3}$$

Ou seja, para cada atributo de E_i, seleciona-se o maior valor de similaridade com atributos de E_j e obtém-se a média desses valores.

A Tabela 1 apresenta os valores de similaridade de atributo S_a encontrados para cada par de entidades. Os valores destacados em negrito representam, para cada entidade do modelo 1, qual entidade do modelo 2 obteve o maior escore.

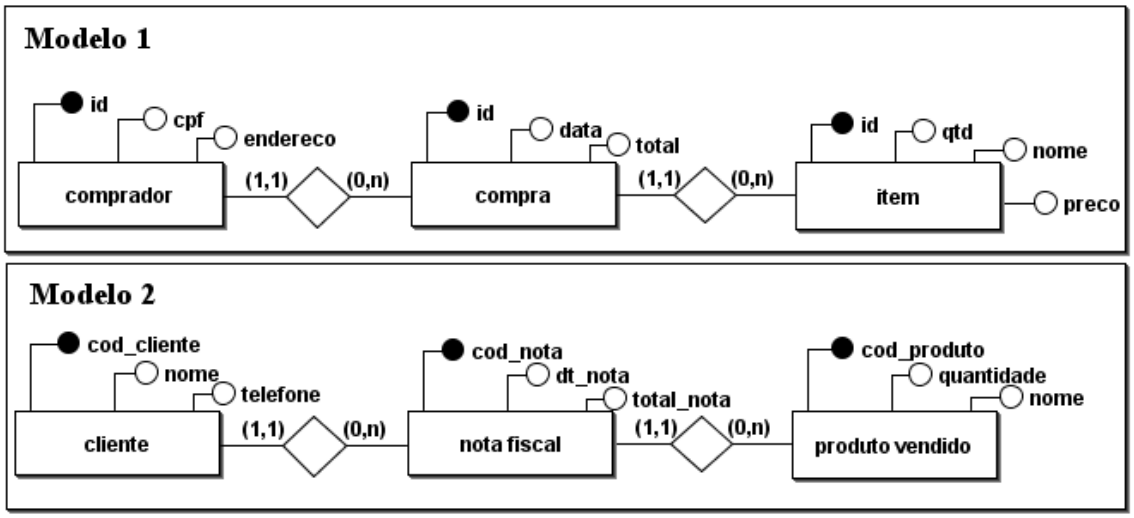

Figura 4. Modelos Conceituais Usados como Exemplo

3.5. Etapa 5: Similaridade das estruturas

A similaridade estrutural entre duas entidades é calculada com base na informação da vizinhança, ou seja, as entidades com as quais exista algum relacionamento direto.

O Algoritmo 1 descreve como esse cálculo é realizado. Para compreender o pseudocódigo, considere que $rel(E_i)$ seja uma função que retorne a lista de entidades que possuem relacionamentos com E_i, e $card(E_i, E_j)$ retorne as informações de cardinalidade que existem entre E_i e E_j. Em suma, dadas as entidades E_i e E_j, o algoritmo encontra, para cada entidade E_x que possua relacionamento com E_i, a melhor entidade correspondente E_y, sendo que E_y deve necessariamente possuir relacionamento com E_j. A melhor entidade E_y é escolhida levando em consideração o maior valor (abordagem otimista) entre a similaridade linguística S_n e a similaridade de atributo (S_a). Além disso, o cálculo também analisa se a cardinalidade entre E_i e E_x é equivalente à cardinalidade entre E_j e E_y. Os pesos θ_1 e θ_2 denotam a importância da cardinalidade e das similaridades S_n e S_a, respectivamente.

Algorithm 1 Algoritmo de similaridade estrutural

$score \leftarrow 0$
for each $E_x \in E_i$ **do**
  $temp\_array \leftarrow \{\}$
  for each $E_y \in E_j$ **do**
    $v_1 \leftarrow max(S_n(E_x, E_y), S_a(E_x, E_y))$
    if $card(E_i, E_x) = card(E_j, E_y)$ **then**
      add $((v_1 \times \theta_1) + (1 \times \theta_2))$ into $temp\_array$
    else
      add $((v_1 \times \theta_1) + (0 \times \theta_2))$ into $temp\_array$
    end if
  end for
  score += max(temp_array)
end for
return $\frac{score}{|rel(E_i)|}$

A Tabela 1 apresenta os valores de similaridade estrutural S_e encontrados para cada par de entidades. Os valores destacados em negrito representam, para cada entidade do modelo 1, qual entidade do modelo 2 obteve o maior escore.

3.6. Etapa 6: Cálculo Final

Nessa etapa, os diferentes valores de similaridade entre cada par de entidade são agregados em um único valor. Três pesos são utilizados (α, β, γ), sendo que eles obedecem às seguintes condições:

- $\alpha + \beta + \gamma = 1$
- $\alpha \geq \beta \geq \gamma$

Como trabalha-se com uma abordagem otimista, definiu-se que o maior peso será usado para o maior valor de similaridade, o segundo maior peso para o segundo maior valor de similaridade, e o peso mais baixo para o valor restante.

caso	entidade 1	entidade 2	entidade 3	escore
1	Comprador (CPF,ID, Endereco)	Compra (ID, Data, Total)	Item (ID, Nome, Qtd, Preco)	100%
2	**Pessoa** (CPF,ID, Endereco)	Compra (ID, Data, Total)	Item (ID, Nome, Qtd, Preco)	98%
3	Pessoa (**RG**,ID, Endereco)	Compra (ID, Data, Total)	Item (ID, Nome, Qtd, Preco)	94%
4	Pessoa (RG,ID, Endereco)	**Doacao** (ID, Data, Total)	Item (ID, Nome, Qtd, Preco)	93%
5	Pessoa (RG,ID, Endereco)	Doacao (ID, **Doador**, Total)	Item (ID, Nome, Qtd, Preco)	85%
6	Pessoa (RG,ID, Endereco)	Doacao (ID, Doador, Total)	(**Instituicao** (ID, Nome, Qtd, Preco)	84%
7	Pessoa (RG,ID, Endereco)	Doacao (ID, Doador, Total)	Instituicao (ID, Nome, **orgao**, Preco)	81%
8	Pessoa (RG,ID, **renda**)	Doacao (ID, Doador, Total)	Instituicao (ID, Nome, orgao, Preco)	77%
9	Pessoa (RG,ID, renda)	Doacao (ID, Doador, **valor**)	Instituicao (ID, Nome, orgao, Preco)	69%
10	Pessoa (RG,ID, renda)	Doacao (ID, Doador, valor)	Instituicao (ID, Nome, orgao, **local**)	62%

Tabela 2. Similaridade entre Modelos que Sofreram Alterações Graduais

Para cada entidade do modelo 1, escolhe-se o maior escore agregado com as entidades do modelo 2. Finalmente, o escore de similaridade final entre dois modelos é dados pelo média aritmética dos melhores escores agregados de entidades do modelo 1.

A Tabela 1 apresenta o escore calculado entre cada par de entidades, usados como pesos $\alpha = 0.6$, $\beta = 0.3$ e $\gamma = 0.1$. A tabela também exibe o escore final de similaridade entre os modelos.

4. Experimentos

Esta seção relata experimentos realizados com o objetivo de avaliar o uso da abordagem proposta para a avaliar as soluções apresentadas para tarefas de modelagem conceitual. Para a detecção de similaridade foram usados os seguintes pesos: $\theta_1 = 0.2$, $\theta_2 = 0.8$, $\alpha = 0.6$, $\beta = 0.3$, $\gamma = 0.1$.

No primeiro experimento, o objetivo é analisar como o algoritmo se comporta quando um modelo conceitual sofre alterações graduais. A Tabela 2 exibe as entidades dos modelos usados, descritos em uma notação textual simplificada.

O primeiro caso se refere ao modelo 1 usado na Figura 1, contendo os mesmos

relacionamentos e cardinalidades. Os demais casos são variações deste modelo, sendo que um caso é levemente diferente em relação ao caso anterior, e essa diferença está demarcada em negrito. A última coluna exibe o escore de similaridade obtido quando o modelo é comparado com um modelo padrão, que é equivalente ao modelo descrito no caso 1. Um último caso foi avaliado (caso 11, não descrito na tabela), onde a mudança realizada foi a inversão da cardinalidade dos relacionamentos em relação ao caso 10. Nessa situação, o escore obtido foi 50%.

Como se pode ver, a comparação do modelo com ele próprio atinge o escore de similaridade de 100%. A similaridade diminui gradativamente, conforme as alterações aumentam. Por exemplo, os modelos representados nos casos 1 e 10 são completamente distintos, e representam realidades distintas, e essa distinção aparece em um escore de similaridade mais baixo. É importante destacar que o objetivo da abordagem não é necessariamente gerar escores baixos na comparação de modelos distintos, mas gerar escores mais baixos do que modelos que possuem maior semelhança. Mesmo que o escore do último caso seja superior a 50%, a redução gradual no escore satisfaz o objetivo proposto.

No segundo experimento, foram usados modelos conceituais criados por alunos de uma disciplina de Fundamentos de Bancos de Dados, ofertada em 2021/02, pelo Departamento de Linguagens e Sistemas de Computação da UFSM. Para a criação dos modelos, os alunos se basearam em uma descrição textual de um sistema de *streaming* de vídeos.

O modelo criado pelo professor da disciplina (gabarito) foi utilizado para realizar a correção automatizada da tarefa. Ou seja, a detecção de similaridade entre um modelo e o gabarito foi usada para estimar a correção do problema.

Ao todo, 22 modelos foram avaliados, sendo que o escore mais baixo e mais alto obtidos foram respectivamente 81% e 98%. A Figura 5 exibe estes dois modelos, juntamente com o gabarito.

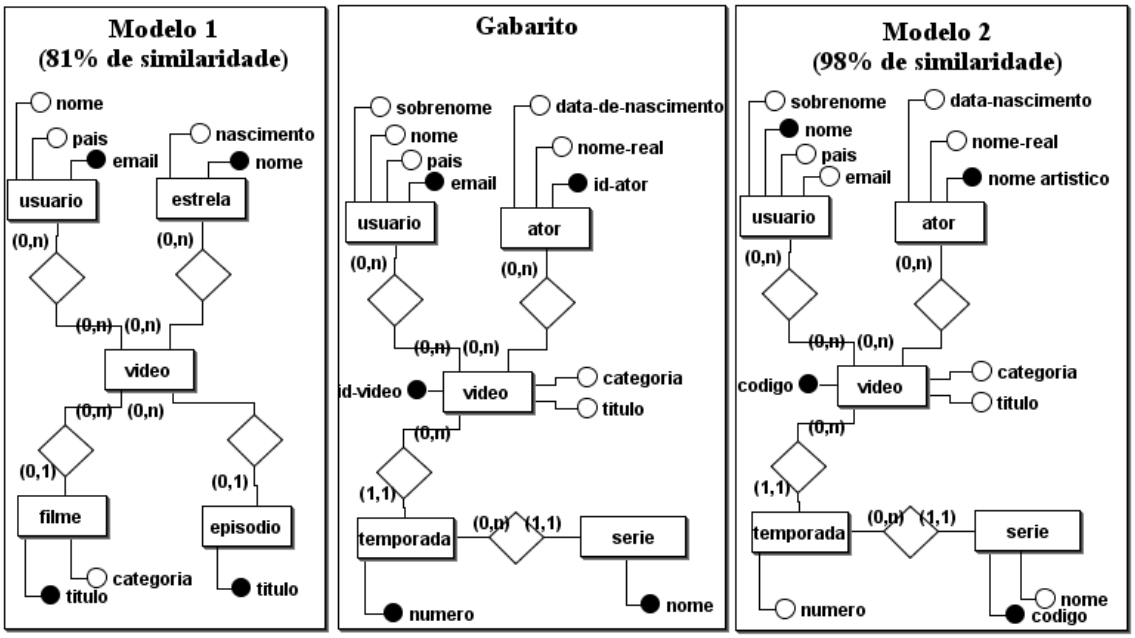

Figura 5. Modelos conceituais Avaliados Automaticamente perante um Gabarito

Percebe-se que o modelo 2 é visivelmente mais semelhante ao gabarito, com pequenas mudanças na definição dos atributos. Como a versão atual da ferramenta não analisa identificadores, o escore não foi afetado por esse fator. Já o modelo 1, apesar de claramente refletir o problema proposto, deixa de representar os conceitos referentes à temporada e série, o que resultou em um escore final mais baixo.

5. Considerações Finais

Este artigo apresentou uma proposta de algoritmo otimista para a detecção da similaridade entre dois modelos conceituais de bancos de dados. Foram apresentados dois experimentos que avaliam o comportamento do algoritmo para o problema de detecção de plágio e correção de tarefas. Os escores altos encontrados refletem o caráter otimista do algoritmo.

Cabe salientar que o objetivo da ferramenta não é automatizar as tarefas de correção e detecção de plágio, mas servir como assistente para que uma análise mais aprofundada seja realizada. A proposta também pode ser empregada em ambientes virtuais de aprendizagem como um mecanismo de autocorreção, em que o aluno realiza envios sucessivos de uma tarefa, usando o escore de similaridade como *feedback* que mede o progresso do aprendizado.

Como trabalhos futuros, pretende-se incorporar outros elementos à linguagem específica de domínio proposta, e usar esses elementos dentro do processo de detecção de similaridade.

Referências

Algergawy, A., Nayak, R., and Saake, G. (2010). Element similarity measures in xml schema matching. *Information Sciences*, 180(24):4975–4998.

Bergroth, L., Hakonen, H., and Raita, T. (2000). A survey of longest common subsequence algorithms. In *Proceedings Seventh International Symposium on String Processing and Information Retrieval. SPIRE 2000*, pages 39–48.

Dimitrieski, V., Čeliković, M., Aleksić, S., Ristić, S., Alargt, A., and Luković, I. (2015). Concepts and evaluation of the extended entity-relationship approach to database design in a multi-paradigm information system modeling tool. *Computer Languages, Systems & Structures*, 44:299–318.

Jean-Mary, Y. R., Shironoshita, E. P., and Kabuka, M. R. (2009). Ontology matching with semantic verification. *Journal of Web Semantics*, 7(3):235–251.

Shvaiko, P. and Euzenat, J. (2011). Ontology matching: state of the art and future challenges. *IEEE Transactions on knowledge and data engineering*, 25(1):158–176.

Tahraoui, M. A., Pinel-Sauvagnat, K., Laitang, C., Boughanem, M., Kheddouci, H., and Ning, L. (2013). A survey on tree matching and xml retrieval. *Computer Science Review*, 8:1–23.