

Seleção Automática de Exemplos de Treino para um Método de Deduplicação de Registros baseado em Programação Genética

Gabriel Silva Gonçalves, Moisés G. de Carvalho,
Alberto H. F. Laender, Marcos André Gonçalves

Departamento de Ciência da Computação, Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte – MG – Brasil

{gabriels, moisesgc, laender, mgoncalv}@dcc.ufmg.br

Abstract. *Recently, machine learning techniques have been used to solve the record deduplication problem. However, they require examples, manually generated in most cases, for training purposes. This uneases the use of such techniques because of the cost required to create the set of examples. In this paper, we propose an approach based on a deterministic technique to automatically suggest training examples for a deduplication method based on genetic programming. Our experiments using synthetic datasets show that using only 15% of the examples suggested by our approach, it is possible to achieve results in terms of F1, equivalent to those obtained when using all the examples, leading to savings in training time of up to 85%.*

Resumo. *Recentemente, técnicas de aprendizagem de máquina vêm sendo utilizadas para resolver o problema de deduplicação de registros. Porém, elas exigem exemplos, normalmente gerados de forma manual, para seu treinamento. Isto torna a utilização dessas técnicas pouco viável na maioria das vezes, devido ao custo exigido para se criar o conjunto de exemplos. Neste artigo, propomos uma abordagem, baseada em uma técnica determinística, para sugerir automaticamente exemplos de treino para um método de deduplicação baseado em programação genética. Nossos experimentos, utilizando conjuntos de dados sintéticos, mostram que com apenas 15% dos exemplos sugeridos por nossa abordagem, é possível atingir resultados, em termos de F1, equivalentes aos obtidos quando se utilizam todos os exemplos encontrados, representando uma economia no tempo de treino de até 85%.*

1. Introdução

Devido ao grande aumento do volume de informação disponível em meios digitais, administradores de grandes repositórios de dados, tais como bibliotecas digitais e bancos de dados de grandes corporações, vêm encontrando problemas em manter a qualidade dos dados disponíveis em seus repositórios. Uma vez que os repositórios de dados podem ser construídos e complementados através da integração com outras fontes de dados, é possível que ocorram inconsistências, permitindo a geração indevida de dados duplicados, resultando em repositórios de dados “sujos”.

Atualmente, é possível estabelecer uma relação entre a qualidade dos dados presentes nos sistemas de uma organização e sua capacidade de prover serviços de

qualidade a seus clientes. A decisão de manter os repositórios de dados “sujos” vai além de questões técnicas, como performance e qualidade dos sistemas que utilizam esses dados. Além de esforços técnicos, também são necessárias mudanças culturais e de gerenciamento dos dados.

A manutenção de repositórios de dados “sujos” pode acarretar diversos problemas. O desempenho do sistema gerenciador de banco de dados possivelmente será afetado, uma vez que dados adicionais sem utilidade demandam um maior processamento, exigindo mais tempo para responder consultas simples feitas pelos usuários. A qualidade das informações extraídas do repositório também será prejudicada, pois a presença de réplicas e inconsistências pode gerar distorções em relatórios, levando à tomada de decisões incorretas. Além disso, é esperado que ocorra um aumento de custos operacionais, já que uma elevação (desnecessária) no volume de dados acarreta maiores investimentos em mídias de armazenamento e poder de processamento computacional para manter os tempos de resposta aos usuários em níveis aceitáveis.

O problema de detectar e remover entradas duplicadas em repositórios é conhecido como deduplicação de registros [Koudas et al. 2006], mas também é denominado na literatura de limpeza de dados [Chaudhuri et al. 2003], associação de registros [Bhattacharya & Getoor 2004, Fellegi & Sunter 1969, Koudas et al. 2006] e casamento de registros [Verykios et al. 2003]. Mais especificamente, a deduplicação de registros em repositórios de dados consiste na identificação e remoção de registros que se referem ao mesmo objeto ou entidade do mundo real, ainda que apresentem estilos de escrita, grafias, tipos de dados ou representações de esquemas diferentes. Recentemente, tem havido um grande investimento por parte de empresas e instituições governamentais no desenvolvimento de métodos realmente eficientes para remoção de réplicas em grandes repositórios de dados [Bell & Dravis 2006, Wheatley 2004]. Entretanto, a deduplicação de registros é uma tarefa bastante complexa, exigindo um tratamento que requer muito poder e tempo de processamento devido à grande quantidade de comparações de registros necessárias. Logo, os métodos propostos para deduplicação devem procurar atingir seus objetivos da forma mais eficiente possível.

Recentemente, de Carvalho et al. (2008a, 2008b) apresentaram uma abordagem inovadora para identificar registros duplicados em repositórios de dados, recorrendo a uma técnica de aprendizagem de máquina conhecida como Programação Genética (PG) [Banzhaf et al. 1998, Koza 1992]. Através dessa abordagem, registros são deduplicados utilizando evidências extraídas do conteúdo dos dados, criando uma função de similaridade capaz de apontar quais registros do repositório são réplicas.

Entretanto, apesar dos resultados superiores a outras abordagens encontradas na literatura, técnicas baseadas em aprendizagem de máquina geralmente necessitam de uma etapa de treino, na qual os exemplos para aprendizado dos padrões de duplicação normalmente são gerados de forma manual. Dessa forma, o custo e tempo necessários para se criar o conjunto de exemplos de treino muitas vezes dificultam a utilização prática dessas técnicas.

Neste artigo, propomos uma abordagem baseada em uma técnica determinística que sugere, de forma automática, exemplos para a etapa de treino do processo de

deduplicação utilizando PG. Inicialmente, verificamos se é realmente necessário utilizar todos os pares de exemplos gerados para a etapa de treino. Realizamos diversos experimentos nos quais a quantidade de exemplos utilizados foi reduzida gradualmente, verificando-se como cada redução afetava a qualidade dos indivíduos gerados ao final do processo de deduplicação. Em seguida, utilizamos um método determinístico para gerar os exemplos para a etapa de treino do processo de deduplicação utilizando PG e analisando a viabilidade de se selecionar os exemplos de forma automática.

Através dos resultados obtidos, mostramos que é possível utilizar uma quantidade reduzida de exemplos de treino sem afetar a qualidade das soluções finais, reduzindo de forma significativa o tempo de treinamento necessário. Além disso, essa seleção pode ser feita de forma automática, sem a necessidade de gerar manualmente um conjunto de treino.

O restante do artigo está organizado da seguinte maneira. Na Seção 2, discutimos trabalhos relacionados. Na Seção 3, apresentamos uma visão geral do processo de deduplicação de registros utilizando PG. Na Seção 4, descrevemos como funciona o processo de seleção automática de exemplos de treino. Na Seção 5, apresentamos nossos experimentos e discutimos os resultados obtidos. Finalmente, na Seção 6, apresentamos algumas considerações finais e possíveis trabalhos futuros.

2. Trabalhos Relacionados

A deduplicação de registros é um tópico de pesquisa que tem atraído bastante interesse em bancos de dados e áreas relacionadas. Conforme já ressaltado, a ocorrência de réplicas em repositórios de dados leva a inconsistências que podem afetar severamente diversos serviços, causando prejuízos para empresas e instituições governamentais.

Na tentativa de resolver essas inconsistências, alguns trabalhos propõem a criação de funções de similaridade que combinem as informações contidas nos repositórios para identificar quando um par de registros são ou não réplicas. Elmagarmid et al. (2007) classificam essas abordagens em duas categorias: (1) ad-hoc, abordagens que geralmente dependem do conhecimento de um domínio ou de métricas de distância específicas (por exemplo, para cadeias de caracteres), e (2) baseadas em treino, abordagens que dependem de algum tipo de treinamento, supervisionado ou semi-supervisionado, para identificação de réplicas, como abordagens probabilísticas e de aprendizagem de máquina.

Fellegi e Sunter (1969) propuseram uma elaborada abordagem estatística para lidar com o problema de combinação de evidências. O método requer a definição de dois limiares para identificação de réplicas. Se o valor de similaridade estiver acima do Limiar de Identificação Positiva, os registros são considerados réplicas; se estiver abaixo do Limiar de Identificação Negativa, os mesmos são considerados não-réplicas; e se o valor estiver entre esses dois limiares, os registros são classificados como “possíveis réplicas”, exigindo um julgamento humano para classificá-los. Por ser uma técnica determinística, não é necessário fornecer exemplos para a etapa de treino, como ocorre nas abordagens de aprendizagem de máquina. Entretanto, esse método tem a desvantagem de exigir do usuário a definição manual dos limiares, tarefa geralmente não trivial, uma vez que esses valores dependem de características do repositório que

será deduplicado. O Febrl¹ é uma das ferramentas que implementam esse método.

As técnicas de aprendizagem de máquina, por sua vez, necessitam de uma pequena porção de dados para treino. Estes dados devem apresentar as mesmas características do conjunto a ser deduplicado, tornando possível uma generalização das soluções obtidas para os dados restantes da base original. O maior problema desse tipo de abordagem é o custo para criação do conjunto de treino, uma tarefa que pode ser pouco viável em muitos casos.

Em [Bilenko et al. 2003] e [Bilenko & Mooney 2003], os autores apresentam o Marlin, um sistema que utiliza uma técnica de aprendizagem de máquina para melhorar as funções de similaridade utilizadas na comparação de atributos dos registros e a forma como as evidências, vetores de termos utilizados para treino de um classificador baseado em SVM (*Support Vector Machines*), são combinadas.

Em [de Carvalho et al. 2006], os autores utilizam PG para gerar uma combinação de evidências melhor que a simples soma utilizada pelo método de Fellegi e Sunter (1969). Em [de Carvalho et al. 2008a], é proposta uma nova abordagem baseada em PG para encontrar a melhor combinação de evidências em um arcabouço genérico independente de qualquer outra técnica. Por fim, [de Carvalho et al. 2008b] apresentam um detalhado estudo experimental para mostrar como a seleção dos parâmetros do processo de PG pode afetar o desempenho do método de deduplicação, sugerindo que a escolha de valores mais adequados conduzem o processo a soluções mais rápidas e eficientes.

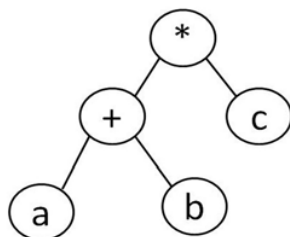
Tanto em [de Carvalho et al. 2006] quanto em [de Carvalho et al. 2008a], os autores separam o conjunto de registros em quatro arquivos de tamanhos iguais, selecionando um deles para treino e os demais para avaliação. Todos os registros do arquivo de treino são comparados e utilizados na criação do conjunto de exemplos de treino. Dependendo do tamanho do repositório a ser deduplicado, esse conjunto de treino pode ser muito grande, fazendo com que o processo de deduplicação se torne ainda mais custoso.

Neste artigo, apresentamos os resultados de um estudo experimental que mostra como o tamanho do conjunto de treino do processo de deduplicação de registros utilizando PG influencia na qualidade das soluções geradas ao final do processo. Além disso, propomos o uso de um método determinístico para gerar o conjunto de treino para o processo de deduplicação, permitindo estudar a viabilidade de se selecionar os exemplos para treino de forma automática, de modo a eliminar a necessidade de se gerar manualmente um conjunto de treino.

3. Visão Geral do Processo de Deduplicação usando Programação Genética

Programação genética é uma das mais conhecidas e utilizadas técnicas de programação evolucionária. Essa técnica pode ser vista como uma heurística adaptativa cujas idéias básicas são originadas das propriedades dos genes e da seleção natural. É uma evolução direta dos programas ou algoritmos usados para aprendizado indutivo (supervisionado), inicialmente aplicados em problemas de otimização.

¹ Freely Extensible Biomedical Record Linkage – <http://sourceforge.net/projects/febrl>



$$\text{árvore}(a,b,c) = (a + b) * c$$

Figura 1. Exemplo de uma Função mapeada como Árvore

Uma das principais características das técnicas evolucionárias é a capacidade de se trabalhar com problemas multiobjetivos, normalmente modelados como restrições do ambiente durante o processo evolucionário [Banzhaf et al. 1998]. Essas abordagens também são conhecidas pela capacidade de se procurar por soluções em grandes (e possivelmente infinitos) espaços de busca, nos quais a solução ótima pode ser desconhecida, geralmente fornecendo respostas bem próximas do ótimo [Banzhaf et al. 1998, Koza 1992].

Um dos principais aspectos que diferencia a PG das demais técnicas evolucionárias (por exemplo, algoritmos genéticos e sistemas evolucionários) é a forma como ela representa os conceitos e interpreta o problema – como um programa de computador, sendo que os dados são vistos e manipulados dessa forma. Além disso, as estruturas dos programas em evolução são flexíveis, ou seja, não apresentam limitações de tamanho.

Em [de Carvalho et al. 2008a, 2008b], é utilizada uma representação baseada em árvores (*indivíduos*) para representar as soluções para o problema e construir as funções de combinação de evidências, conforme ilustrado na Figura 1. Ao utilizar essa representação, deve-se definir inicialmente um conjunto de terminais e funções. Os terminais são entradas, constantes ou nós com aridade zero, também denominados folhas, enquanto o conjunto de funções é constituído por operadores, declarações e funções básicas ou definidas pelo usuário, utilizadas no processo para manipular os valores dos terminais [Koza 1992]. Os nós folhas se encontram ao final dos ramos das árvores, enquanto as funções são colocadas em seus nós internos, como podemos ver na Figura 1.

Para a deduplicação de registros, são utilizadas funções que combinam evidências, sendo que cada evidência E é formada por um par $\langle \text{atributo}, \text{função de similaridade} \rangle$ que representa o uso de uma função de similaridade específica sobre valores de um determinado atributo do repositório de dados.

Por exemplo, para deduplicar a tabela de um banco de dados relacional com os atributos *nome*, *sobrenome*, *idade* e *endereço*, utilizando uma determinada função de similaridade (por exemplo, Jaro-Winkler (JW) [Koudas et al. 2006]), teremos a seguinte lista de evidências: $E_1 \langle \text{nome}, JW \rangle$, $E_2 \langle \text{sobrenome}, JW \rangle$, $E_3 \langle \text{idade}, JW \rangle$ e $E_4 \langle \text{endereço}, JW \rangle$.

Para este exemplo, uma função simples (F_s) poderia ser uma combinação linear da seguinte forma

$$F_s(E_1, E_2, E_3, E_4) = E_1 + E_2 + E_3 + E_4,$$

enquanto uma função mais complexa (F_c) poderia ser da forma

$$F_c(E_1, E_2, E_3, E_4) = E_1 * (E_2 / (E_3 \exp E_4)).$$

Para modelar essas funções em formato de árvore, cada evidência é representada por uma folha (através de valores reais normalizados entre 0,0 e 1,0), enquanto os nós internos representam as operações que são aplicadas às folhas, operações matemáticas (por exemplo, +, -, *, /, exp) que manipulam esses valores.

Por fim, a árvore modelada é avaliada automaticamente, viabilizando o uso dessa técnica. Durante o processo evolucionário da PG, todas as árvores geradas são testadas em repositórios de dados previamente avaliados, onde as réplicas já foram devidamente identificadas, permitindo avaliar a capacidade de cada árvore reconhecer os pares de registros que sejam réplicas verdadeiras.

As entradas da árvore são formadas por instâncias de evidências extraídas dos dados manipulados. Já a saída consiste em um valor real (resultado da operação codificada na árvore) que é comparado com um limiar de identificação de réplicas da seguinte forma: se for superior a esse limiar, os registros são considerados réplicas; caso contrário, os registros são considerados distintos. Essa abordagem de classificação respeita as propriedades de transitividade das réplicas (se A for réplica de B e B for réplica de C, então A será réplica de C).

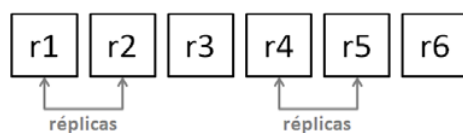
Após a comparação de todos os pares de registros gerados, é computado o número total de identificações de réplicas corretas e incorretas. Essa informação é utilizada posteriormente pela função de aptidão, componente responsável pela avaliação dos indivíduos gerados durante todo o processo evolucionário. Para os nossos experimentos, escolhemos a métrica F1 como função de aptidão, que expressa, através de um único valor, o quão bem um indivíduo específico consegue realizar a tarefa de identificação de réplicas. Essa métrica combina harmonicamente as tradicionais métricas de precisão e revocação utilizadas em avaliações de sistemas de recuperação de informação [Baeza-Yates & Ribeiro-Neto 1999], sendo definida da seguinte forma:

$$\text{Precisão} = \frac{\text{QuantidadeDeParesDeRéplicasIdentificadosCorretamente}}{\text{QuantidadeDeParesDeRéplicasIdentificados}}$$

$$\text{Revocação} = \frac{\text{QuantidadeDeParesDeRéplicasIdentificadosCorretamente}}{\text{QuantidadeDeParesDeRéplicasExistentes}}$$

$$F1 = \frac{2 \times \text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}}$$

Durante o processo evolucionário, os indivíduos são manipulados e modificados através de operações genéticas como avaliação, reprodução, cruzamento (*crossover*), seleção e mutação, em um processo que tenta gerar indivíduos melhores em cada geração subsequente. Maiores detalhes sobre PG podem ser encontrados em [Banzhaf et al. 1998] e [Koza 1992].



Pares de registros possíveis:

(r1,r2), (r1,r3), (r1,r4), (r1,r5), (r1,r6)

(r2,r3), (r2,r4), (r2,r5), (r2,r6)

(r3,r4), (r3,r5), (r3,r6)

(r4,r5), (r4,r6)

(r5, r6)

Pares de registros positivos (em negrito)

Pares de registros negativos (em itálico)

Figura 2. Exemplo de Pares de Registros Positivos e Negativos

4. Seleção Automática de Exemplos de Treino

Nesta seção, apresentaremos a nossa abordagem para seleção automática de exemplos para a etapa de treino do processo de deduplicação utilizando PG. Primeiramente, apresentaremos algumas definições para o entendimento do processo de seleção, seguindo-se uma explicação mais detalhada sobre o funcionamento da nossa abordagem.

4.1 Definição de Pares de Registros

No processo de deduplicação utilizando PG, os registros são agrupados em pares, de forma que possam ser comparados através de funções de similaridade pré-definidas, na tentativa de se apontar quais registros do repositório são réplicas. Dessa forma, quando falamos de exemplos de treino, é necessário identificar *pares de registros positivos* e *pares de registros negativos*.

Um *par de registros positivo* é formado por dois registros que fazem referência ao mesmo objeto ou entidade do mundo real, ou seja, após a comparação, são apontados como réplicas um do outro. Por outro lado, um *par de registros negativo* é constituído por dois registros que não fazem referência ao mesmo objeto do mundo real, não sendo apontados como réplicas um do outro. Esses dois tipos de exemplo auxiliam o método de deduplicação utilizando PG na compreensão de o que é réplica e o que não é, dentro do repositório de dados.

Ao deduplicar um conjunto de dados, cada registro deve ser comparado com os demais. Logo, para um repositório contendo n registros, devem ser realizadas $\frac{n*(n-1)}{2}$ comparações entre dois registros. O divisor é igual a dois pois, na realidade, apenas metade das comparações é realizada, já que dois registros nunca são comparados mais de uma vez. A maioria das comparações corresponderão a não-réplicas (pares de registros negativos), uma vez que o número máximo de pares replicados é geralmente menor que a quantidade de registros no repositório. A Figura 2 ilustra exemplos de pares de registros positivos e negativos em um repositório composto por seis registros.

4.2 Seleção Automática de Exemplos

Em [de Carvalho et al. 2008a, 2008b], após a criação do conjunto de registros utilizando o gerador de dados sintéticos (ou mesmo nos experimentos que utilizam dados reais), os

exemplos são divididos igualmente em quatro arquivos, sendo um deles utilizado para a etapa de treino e os demais para a etapa de avaliação. Todos os pares de registros gerados para treino são comparados e classificados como réplicas ou não-réplicas, resultando em uma lista de exemplos para a etapa de treino, que é a mesma utilizada em todas as gerações e rodadas dos experimentos.

Ao utilizar o gerador de dados sintéticos, a própria ferramenta informa os pares de registros duplicados existentes. Essa informação é importante para a avaliação dos indivíduos gerados durante a etapa de treino, uma vez que a função de aptidão F1 utiliza informações que devem ser conhecidas previamente, como o número total de pares de registros duplicados no repositório de dados.

Entretanto, ao utilizar repositórios de dados reais, informações como essa são desconhecidas, exigindo-se que o usuário classifique manualmente (em réplicas e não-réplicas) todos os pares de registros do conjunto de exemplos para a etapa de treino do processo de deduplicação utilizando PG.

Para evitar tal procedimento, a nossa abordagem utiliza uma técnica determinística para selecionar automaticamente um subconjunto desses exemplos para ser utilizado na etapa de treino. Primeiramente, o método de Fellegi e Sunter (1969), descrito na Seção 2, é utilizado para deduplicar o conjunto de treino (que pode ser o mesmo utilizado nos experimentos iniciais), gerando ao final uma relação dos pares de registros que são réplicas e que não são réplicas. Em seguida, extraímos aleatoriamente dessa relação diferentes quantidades de pares de registros e montamos um novo conjunto de exemplos para ser utilizado na etapa de treino. Por fim, executamos novamente o processo de deduplicação utilizando PG, mas usando como entrada o conjunto de dados de treino gerado pelo método de Fellegi e Sunter.

Assim, facilitamos a criação do conjunto de exemplos de treino e tornamos a utilização prática da técnica de PG para deduplicação de registros mais viável, removendo a necessidade de qualquer interação humana no processo de seleção de exemplos para o treino do processo de deduplicação de registros utilizando PG. É importante ressaltar que, embora o método utilizado na primeira etapa da nossa abordagem tenha sido o de Fellegi e Sunter, é possível utilizar outros métodos determinísticos de classificação, como o k-means [Gu & Baxter 2006].

5. Experimentos

Nesta seção, apresentamos os resultados dos experimentos realizados utilizando a nossa abordagem de seleção automática de exemplos de treino para deduplicação de registros utilizando PG.

Na primeira parte do nosso estudo experimental, realizamos três experimentos distintos, variando a quantidade dos pares de registros utilizados na etapa de treino. Primeiramente, reduzimos a quantidade de pares de registros positivos gradualmente e mantivemos a totalidade dos pares de registros negativos. Em seguida, fizemos a redução gradual da quantidade de pares de registros negativos, mantendo a totalidade dos pares de registros positivos. Por fim, reduzimos tanto a quantidade de pares de registros positivos quanto de negativos. Os valores dos percentuais de redução foram

escolhidos de forma empírica, após a realização de experimentos iniciais.

A intenção desses três experimentos é mostrar como a escolha da quantidade de exemplos (pares de registros positivos e negativos) utilizados na etapa de treino impacta a qualidade dos indivíduos gerados ao final do processo de deduplicação de registros. Além disso, procuramos verificar se é realmente necessário utilizar todos os exemplos gerados para treino, como foi feito em [de Carvalho et al. 2008a, 2008b]. Os resultados dessa avaliação permitem a sugestão de configurações para seleção de dados para treino, possibilitando a identificação de réplicas em repositórios de dados de forma mais eficiente, sem que a qualidade das soluções geradas sejam prejudicadas.

Na segunda parte do nosso estudo experimental, realizamos o processo de deduplicação de registros utilizando exemplos de treino gerados pelo método determinístico de Fellegi e Sunter (1969), disponível na ferramenta Febrl. A idéia desse experimento é avaliar se os exemplos gerados automaticamente são bons exemplos de treino, avaliando-os através da métrica F1. A seleção dos exemplos de treino de forma automática torna o processo de deduplicação utilizando técnicas de aprendizagem de máquina mais acessíveis.

A configuração dos parâmetros da PG e a relação de evidências utilizadas nesse trabalho são as mesmas estabelecidas para experimentação em [de Carvalho et al. 2008a]. Todos os experimentos foram realizados utilizando três computadores com a seguinte configuração: processador Pentium Core 2 Quad de 2 GHz, com 4 GB RAM DDR2 de memória principal e HD SATA de 320 GB, rodando o sistema operacional FreeBSD 7.1 64-Bits e utilizando Python 2.5.1.

5.1 Conjunto de Dados Experimental

Neste artigo, utilizamos um conjunto de dados criado através do SDG [Christen 2005], um gerador de dados sintéticos disponível na ferramenta Febrl. Com esse gerador, é possível criar conjuntos de dados contendo nomes (baseados em tabelas de frequência para nomes e sobrenomes), endereços (baseados em tabelas de frequência para localidades, códigos postais, números de ruas, etc.), números telefônicos e identificadores numéricos para pessoas (como o número de seguridade social). Uma vez que dados reais não são facilmente disponibilizados para utilização em experimentos, devido a restrições de privacidade e confidencialidade, recorreremos a conjuntos de dados sintéticos para nossos experimentos.

Os dados gerados pelo SDG são similares aos frequentemente encontrados em registros de dados médicos pessoais. Primeiramente, a ferramenta cria um conjunto de dados contendo apenas registros originais. Em seguida, o SDG gera réplicas desses registros, realizando modificações como inserção, remoção e substituição de caracteres, além de troca, remoção, inserção e divisão de palavras, alterações baseadas em características reais de erros. Essas réplicas são então inseridas no conjunto de dados original para a realização dos experimentos de deduplicação de registros.

Para a realização dos experimentos, criamos um conjunto de dados sintéticos contendo 2.000 registros distribuídos igualmente em quatro arquivos. Cada arquivo é composto de 300 registros originais e 200 registros réplicas. Essas réplicas são geradas obedecendo as seguintes restrições: no máximo uma réplica pode ser criada baseada em

um registro original (utilizando uma distribuição uniforme), no máximo uma modificação pode ser feita em cada atributo do registro e no máximo um atributo pode ser alterado no registro todo. Os registros possuem os seguintes atributos: *nome, sobrenome, número da rua, endereço1, endereço2, bairro, código postal, estado, data de aniversário, idade, número telefônico e número de seguridade social*.

Em nossos experimentos, utilizamos uma proporção maior de registros replicados em nossos conjuntos de dados de treino e teste do que a encontrada em cenários reais, como o reportado pelo projeto USIIS², em que a taxa de registros replicados é de aproximadamente 20%. Apesar disso, apenas exemplos representativos para treino, ou seja, aqueles mais úteis para o aprendizado dos padrões de duplicação, são efetivamente utilizados. Tal procedimento também é adotado em [de Carvalho et al. 2008a].

A utilização de dados sintéticos em nossos experimentos permite uma melhor avaliação do impacto na qualidade das soluções finais, resultado das mudanças na quantidade de pares de registros positivos e negativos utilizada na etapa de treino, uma vez que os erros existentes e as características dos registros são conhecidas *a priori*. Todos os experimentos envolvem a realização de dois passos: (1) um arquivo é selecionado para a etapa de treino e (2) o arcabouço de PG testa os resultados da etapa de treino nos três arquivos restantes.

Para todos os experimentos, apresentamos os valores de média e desvio padrão após a execução de dez rodadas, conforme realizado em [de Carvalho et al. 2008b].

5.2 Redução dos Exemplos de Treino

Conforme explicado anteriormente, a intenção desses experimentos é observar como a qualidade dos indivíduos gerados no processo de deduplicação e o tempo gasto para treino são afetados pelas mudanças na quantidade dos dados utilizados no treinamento do processo utilizando PG. Em todos os experimentos realizados, os pares de registros foram selecionados de forma aleatória, nas proporções definidas para cada experimento.

5.2.1 Redução de Pares de Registros Positivos

Neste experimento, mantemos a totalidade dos pares de registros negativos e reduzimos gradualmente a quantidade de pares de registros positivos para utilização na etapa de treino. Os pares de registros positivos são minoria no conjunto de pares utilizados nessa etapa, mas eles influenciam a qualidade dos indivíduos gerados no processo de deduplicação. Os resultados do experimento são apresentados na Tabela 1.

O tempo total gasto na etapa de treino em cada configuração também é apresentado na Tabela 1, sendo útil para efeito de comparação. Os resultados obtidos na fase de teste são apresentados na quarta coluna em diante, indicando o valor médio de F1 e o desvio padrão do melhor indivíduo em cada arquivo de teste (A, B e C). Os resultados dos outros experimentos são apresentados em tabelas com estruturas similares.

² Utah Statewide Immunization Information System –
http://health.utah.gov/phi/brownbag/handouts/2008/USIIS_april.pdf

Tabela 1. Redução de Pares de Registros Positivos – Tempos para Treino, Médias e Desvios Padrões de F1

| % Pares Negativos | % Pares Positivos | Tempo para Treino (min) | (A) Média | (A) Desvio Padrão | (B) Média | (B) Desvio Padrão | (C) Média | (C) Desvio Padrão |
|-------------------|-------------------|-------------------------|-----------|-------------------|-----------|-------------------|-----------|-------------------|
| 100 | 100 | 2278 | 0,994 | 0,009 | 0,997 | 0,008 | 0,995 | 0,011 |
| | 95 | 2460 | 0,996 | 0,004 | 1,000 | 0,000 | 0,997 | 0,003 |
| | 90 | 2602 | 0,997 | 0,003 | 1,000 | 0,000 | 0,997 | 0,003 |
| | 70 | 2208 | 0,996 | 0,006 | 1,000 | 0,000 | 0,998 | 0,001 |
| | 50 | 2472 | 0,996 | 0,002 | 1,000 | 0,000 | 1,000 | 0,000 |
| | 30 | 2454 | 0,995 | 0,003 | 1,000 | 0,000 | 0,999 | 0,001 |
| | 15 | 2485 | 0,994 | 0,004 | 0,998 | 0,004 | 0,998 | 0,004 |
| | 5 | 2551 | 0,988 | 0,010 | 0,991 | 0,010 | 0,989 | 0,014 |
| | 2,5 | 1980 | 0,967 | 0,035 | 0,972 | 0,032 | 0,967 | 0,040 |
| | 1 | 1939 | 0,952 | 0,052 | 0,957 | 0,049 | 0,949 | 0,055 |
| 0 | 2377 | 0,315 | 0,418 | 0,317 | 0,414 | 0,321 | 0,419 | |

Os resultados demonstram que a redução da quantidade de pares de registros positivos utilizados na etapa de treino afeta a qualidade (medida pela métrica F1) dos resultados obtidos na fase de teste. Entretanto, ao utilizar percentuais reduzidos de pares de registros positivos (por exemplo, 2,5 e 1%), ainda é possível obter resultados próximos daqueles obtidos quando todos os pares de exemplos de treino são utilizados, onde os valores médios de F1 são elevados e os desvios padrões são menores, demonstrando que ocorre uma baixa dispersão dos valores de F1 em torno da média.

Além disso, ao utilizar apenas 1% dos pares de registros positivos, por exemplo, obtemos uma economia de tempo para treino de aproximadamente 15%. Entretanto, não foi possível estabelecer uma relação direta entre o percentual de registros positivos e o tempo gasto para treino.

5.2.2 Redução de Pares de Registros Negativos

Neste conjunto de experimentos, estamos lidando com o tipo de par de registros que ocorre com maior frequência no conjunto de treino. Por esse motivo, é importante analisar como a redução desses pares influencia a qualidade dos resultados obtidos na etapa de teste, além dos tempos gastos para treino. Os resultados do experimento são apresentados na Tabela 2.

Os resultados demonstram que a redução da quantidade dos pares de registros negativos influencia a qualidade dos resultados obtidos na etapa de teste de forma mais significativa que a redução dos pares de registros positivos.

Na Tabela 1, podemos observar que os valores médios de F1 para a configuração definida com 1% dos pares de registros positivos e a totalidade dos pares de registros negativos estão próximos de 0,950, enquanto a definição de 5% dos pares de registros negativos e a totalidade dos pares de registros positivos, vide Tabela 2, já leva a resultados inferiores aos obtidos utilizando a primeira configuração. Apesar disso, utilizando uma quantidade reduzida de pares de registros negativos (por exemplo, 15%), já conseguimos obter soluções próximas daquelas obtidas com a totalidade dos exemplos de treino.

Tabela 2. Redução de Pares de Registros Negativos – Tempos para Treino, Médias e Desvios Padrões de F1

| % Pares Positivos | % Pares Negativos | Tempo para Treino (min) | (A) Média | (A) Desvio Padrão | (B) Média | (B) Desvio Padrão | (C) Média | (C) Desvio Padrão |
|-------------------|-------------------|-------------------------|-----------|-------------------|-----------|-------------------|-----------|-------------------|
| 100 | 100 | 2278 | 0,994 | 0,009 | 0,997 | 0,008 | 0,995 | 0,011 |
| | 95 | 2648 | 0,998 | 0,002 | 0,999 | 0,003 | 0,999 | 0,001 |
| | 90 | 2155 | 0,998 | 0,003 | 1,000 | 0,000 | 0,998 | 0,003 |
| | 70 | 1610 | 0,998 | 0,002 | 0,999 | 0,003 | 0,998 | 0,003 |
| | 50 | 1265 | 0,997 | 0,007 | 0,999 | 0,002 | 0,998 | 0,001 |
| | 30 | 721 | 0,993 | 0,009 | 0,999 | 0,003 | 0,994 | 0,011 |
| | 15 | 353 | 0,995 | 0,009 | 0,996 | 0,007 | 0,996 | 0,006 |
| | 5 | 140 | 0,926 | 0,138 | 0,925 | 0,132 | 0,905 | 0,185 |
| | 2,5 | 81 | 0,846 | 0,154 | 0,851 | 0,137 | 0,826 | 0,183 |
| | 1 | 36 | 0,677 | 0,242 | 0,649 | 0,260 | 0,654 | 0,266 |
| | 0 | 20 | 0,163 | 0,213 | 0,163 | 0,221 | 0,143 | 0,286 |

Ao utilizar apenas 15% dos pares de registros negativos, por exemplo, temos uma economia de tempo para treino de aproximadamente 85% com relação ao tempo gasto quando são utilizados todos os pares de registros gerados.

Outro ponto a ser destacado é a existência de uma relação entre a quantidade de pares de registros negativos utilizados e o tempo total gasto para a etapa de treino, ou seja, quanto menos pares de registros negativos são utilizados, menos tempo é gasto nessa etapa, e vice-versa. Essa redução ocorre nesse experimento uma vez que os pares de registros negativos correspondem à maioria dos pares de registros utilizados na etapa de treino. Ao se reduzir 85% dos pares de registros negativos, por exemplo, são desconsiderados 105.868 pares de registros para a etapa de treino, enquanto que a mesma redução percentual de pares de registros positivos corresponde a remoção de apenas 170 pares de registros. Dessa forma, obtém-se uma economia de tempo para treino, visto que a quantidade de pares de registros necessários para essa etapa é consideravelmente reduzida.

5.2.3 Redução de Pares de Registros Positivos e Negativos

Nesta seção, executamos experimentos em que as quantidades de pares de registros positivos e negativos são reduzidas gradualmente e na mesma proporção. Os resultados são apresentados na Tabela 3.

Podemos perceber que a redução das quantidades de pares de registros positivos e negativos utilizadas para treino apresenta uma relação direta com o tempo gasto nessa etapa. Além disso, é possível reduzir consideravelmente a quantidade de pares utilizados para treino e ainda assim obter bons resultados. Para 10% de pares positivos e negativos, por exemplo, a economia de tempo para treino chega a aproximadamente 88%, com perdas na qualidade da deduplicação de apenas 0,6%, 0,7% e 0,3%, nos arquivos de teste A, B e C, respectivamente.

5.3 Deduplicação de Registros Determinística

Através desse experimento, pretendemos validar o nosso processo automático de seleção

Tabela 3. Redução de Pares de Registros Positivos e Negativos – Tempos para Treino, Médias e Desvios Padrões de F1

| % Pares Positivos | % Pares Negativos | Tempo para Treino (min) | (A) Média | (A) Desvio Padrão | (B) Média | (B) Desvio Padrão | (C) Média | (C) Desvio Padrão |
|-------------------|-------------------|-------------------------|-----------|-------------------|-----------|-------------------|-----------|-------------------|
| 100 | 100 | 2278 | 0,994 | 0,009 | 0,997 | 0,008 | 0,995 | 0,011 |
| 50 | 50 | 1322 | 0,995 | 0,002 | 1,000 | 0,000 | 0,999 | 0,001 |
| 25 | 25 | 651 | 0,994 | 0,004 | 0,996 | 0,006 | 0,998 | 0,002 |
| 10 | 10 | 277 | 0,988 | 0,012 | 0,990 | 0,011 | 0,992 | 0,010 |
| 5 | 5 | 143 | 0,936 | 0,142 | 0,942 | 0,132 | 0,924 | 0,189 |
| 2,5 | 2,5 | 66 | 0,941 | 0,051 | 0,948 | 0,051 | 0,952 | 0,042 |
| 1 | 1 | 40 | 0,869 | 0,110 | 0,875 | 0,118 | 0,852 | 0,150 |

Tabela 4. Deduplicação utilizando o Método de Fellegi e Sunter para Geração do Conjunto de Treino – Tempos para Treino, Médias e Desvios Padrões de F1

| % Pares Positivos | % Pares Negativos | Tempo para Treino (min) | (A) Média | (A) Desvio Padrão | (B) Média | (B) Desvio Padrão | (C) Média | (C) Desvio Padrão |
|-------------------|-------------------|-------------------------|-----------|-------------------|-----------|-------------------|-----------|-------------------|
| 10 | 10 | 1324 | 0,983 | 0,015 | 0,983 | 0,018 | 0,975 | 0,018 |
| 5 | 5 | 602 | 0,984 | 0,016 | 0,979 | 0,028 | 0,979 | 0,024 |
| 2,5 | 2,5 | 292 | 0,988 | 0,011 | 0,983 | 0,024 | 0,982 | 0,023 |

de exemplos para a etapa de treino do processo de deduplicação. Em todos os experimentos realizados, os pares de registros sugeridos pelo método determinístico foram selecionados de forma aleatória, nas proporções definidas para cada experimento. Os resultados podem ser vistos na Tabela 4.

Neste experimento, utilizando exemplos sugeridos automaticamente, os indivíduos gerados ao final do processo de deduplicação apresentam elevados valores médios de F1, eliminando a necessidade de identificação manual dos exemplos de treino. Foi possível utilizar 2,5% dos pares de registros positivos e negativos para a etapa de treino sem que ocorresse uma queda perceptível na qualidade dos indivíduos gerados no processo de deduplicação, uma vez que os valores médios de F1, comparando com a configuração que utiliza todos os pares de registros gerados, tiveram uma redução de apenas 0,6%, 1,1% e 1,2%, para os arquivos de teste A, B e C, respectivamente. Logo, é possível utilizar com sucesso nossa abordagem automática de seleção de exemplos de treino do processo de deduplicação de registros utilizando PG.

6. Conclusões e Trabalhos Futuros

A identificação e remoção de réplicas em repositórios de dados é de extrema importância para a manutenção da qualidade das informações disponíveis pelos serviços atuais de armazenamento de dados. Sistemas que dependem da integridade dos dados para oferecer serviços de alta qualidade, como bibliotecas digitais e sistemas de comércio eletrônico, podem ser afetados pela existência de réplicas ou quase-réplicas em seus repositórios. Logo, tem sido feito um grande esforço no desenvolvimento de métodos efetivos e eficientes para remoção de réplicas em grandes repositórios de dados [Bell & Dravis 2006, Wheatley 2004].

Por ser uma tarefa complexa, cujo tratamento requer muito tempo e poder de processamento devido à grande quantidade de comparações de registros necessárias,

técnicas de aprendizagem de máquina têm sido utilizadas com sucesso no tratamento do problema de deduplicação. Entretanto, tais técnicas necessitam de uma etapa de treinamento, na qual exemplos gerados manualmente são usados para aprendizado da tarefa de deduplicação. Essa geração manual dificulta a utilização dessas técnicas em muitas situações, devido ao custo e tempo necessários para se criar o conjunto de exemplos de treino. Neste artigo, propomos uma abordagem, baseada em uma técnica determinística, para sugerir de forma automática exemplos de treino para um método de deduplicação baseado em programação genética, tornando o processo mais eficiente e viável.

Através dos resultados de nosso estudo experimental, demonstramos também ser possível utilizar uma quantidade menor de exemplos sem afetar a qualidade dos resultados obtidos na etapa de teste (geralmente, 2,5% dos pares de registros positivos e negativos já são suficientes para conseguir resultados satisfatórios), reduzindo significativamente o tempo necessário para treinamento.

Como trabalho futuro, pretendemos realizar um estudo experimental utilizando repositórios de dados reais, de diferentes domínios e graus de dificuldade, ajudando a consolidar e estender os resultados obtidos neste trabalho. Além disso, planejamos utilizar outras técnicas para gerar os exemplos a serem usados na etapa de treino, de forma que seja possível comparar diferentes abordagens para a escolha dos exemplos de treino.

Agradecimentos

Este trabalho é parcialmente financiado pelo Instituto Nacional de Ciência e Tecnologia para a Web – INCT Web (processo MCT/CNPq número 550874/2007-0), pelo projeto InfoWeb (processo MCT/CNPq/CT-INFO número 573871/2008-6) e por bolsas individuais concedidas aos autores pela FAPEMIG e CNPq.

Referências

- Baeza-Yates, R. A. & Ribeiro-Neto, B. A. (1999). *Modern Information Retrieval*. ACM Press/Addison-Wesley.
- Banzhaf, W., Nordin, P., Keller, R. E., & Francone, F. D. (1998). *Genetic Programming - An Introduction: on the Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann Publishers.
- Bell, R. & Dravis, F. (2006). Is your data dirty? (And does that matter?). Accenture Whiter Paper - <http://www.accenture.com>.
- Bhattacharya, I. & Getoor, L. (2004). Iterative record linkage for cleaning and integration. In *Proc. of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 11–18.
- Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., & Fienberg, S. (2003). Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23.

- Bilenko, M. & Mooney, R. J. (2003). Adaptive duplicate detection using learnable string similarity measures. In *Proc. of the Ninth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pp. 39–48.
- Chaudhuri, S., Ganjam, K., Ganti, V., & Motwani, R. (2003). Robust and efficient fuzzy match for online data cleaning. In *Proc. of the 2003 ACM SIGMOD Int'l Conf. on Management of Data*, pp. 313–324.
- Christen, P. (2005). Probabilistic data generation for deduplication and data linkage. In *Intelligent Data Engineering and Automated Learning – IDEAL*, Lecture Notes in Computer Science, pp. 109–116. Springer.
- de Carvalho, M. G., Goncalves, M. A., Laender, A. H. F., & da Silva, A. S. (2006). Learning to deduplicate. In *Proc. of the 6th ACM/IEEE-CS Joint Conf. on Digital Libraries*, pp. 41–50.
- de Carvalho, M. G., Laender, A. H. F., Goncalves, M. A., & da Silva, A. S. (2008). Replica identification using genetic programming. In *Proc. of the 23rd Annual ACM Symposium on Applied Computing*, pp. 1801–1807.
- de Carvalho, M. G., Laender, A. H. F., Goncalves, M. A., & Porto, T. C. (2008). The Impact of Parameters Setup on a Genetic Programming Approach to Record Deduplication. In *Anais do Simpósio Brasileiro de Banco de Dados*, pp. 91–105.
- Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. *IEEE Trans. on Knowledge and Data Engineering*, 19(1):1–16.
- Fellegi, I. P. & Sunter, A. B. (1969). A theory for record linkage. *Journal of American Statistical Association*, 66(1):1183–1210.
- Gu, L. & Baxter, R. (2006). Decision models for record linkage. In *Selected Papers from AusDM*, Volume 3755, pp. 146–160. Springer.
- Koudas, N., Sarawagi, S., & Srivastava, D. (2006). Record linkage: similarity measures and algorithms. In *Proc. of the 2006 ACM SIGMOD Int'l Conf. on Management of Data*, pp. 802–803.
- Koza, J. R. (1992). *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. MIT Press.
- Verykios, V. S., Moustakides, G. V., & Elfeky, M. G. (2003). A bayesian decision model for cost optimal record matching. *The Very Large Data Bases Journal*, 12(1):28–40.
- Wheatley, M. (2004). Operation clean data. CIO Asia Magazine, August - <http://www.cio-asia.com>.