Universidade Federal de Pernambuco
Centro de Informática
Graduação em Ciência da Computação

Extraindo informações de referências bibliográficas automaticamente

Trabalho de graduação

Aluno: Max José Lins Timóteo (mjlt@cin.ufpe.br)
Orientadora: Flávia de Almeida Barros (fab@cin.ufpe.br)

Recife
Fevereiro de 2006

Resumo

O objetivo desse trabalho é o desenvolvimento de um sistema que realiza extração de informação de referências bibliográficas de forma automática. Essa tarefa é da área de Extração de Informação, a qual faz parte da área de Recuperação de informação. A solução utilizada no sistema é da área de Inteligência Artificial por ter como base o comportamento humano. Testes realizados no sistema com um conjunto de 50 referências bibliográficas resultaram em uma taxa média de acerto na classificação de palavras e números de 95,55%.

Agradecimentos

Agradeço primeiramente a Deus por ele estar sempre ao meu lado e me dar força para seguir em frente. Também a minha família, que está sempre ao meu lado. Principalmente minha mãe Edleuza, meu pai Gonçalo e meu irmão Marcondys. Sem eles não teria chegado até aqui.

Agradeço aos meus colegas de curso, principalmente Emerson, Ênio e Helder, que quase sempre faziam parte da minha equipe nos projetos da faculdade. E também a outros colegas de faculdade como João Marcelo e Pedro Santos.

Índice

1 - Introdução

1.1 - Objetivos do trabalho

O objetivo principal desse trabalho foi desenvolver um sistema capaz de processar referências bibliográficas, extraindo informações dos dados. Dentre essas informações estão autor, título e ano de publicação. Os algoritmos utilizados estão relacionados à área de extração de informação e também à área de inteligência artificial. A idéia de fazer um sistema que tenha o comportamento semelhante ao humano no que diz respeito à extração de informação também é um dos objetivos do sistema embora não seja o principal. Para isso o sistema utiliza um conjunto de regras que tenta simular regras humanas de classificação de partes de uma referência bibliográfica.

Para simular melhor o comportamento humano de classificação o sistema possui um banco de dados, pois o ser humano utiliza muito a memória quando está lendo, para reconhecer palavras e seus significados. O sistema também armazena novas informações de forma automática. Porém o sistema não gera novas regras automaticamente.

1.2 - Organização do documento

Esta seção explica de forma resumida o que pode ser encontrado nesse documento. No capítulo “Contexto” são comentadas as áreas de inteligência artificial, recuperação de informação e de extração de informação. O capítulo “Arquitetura do sistema” trata as classes do sistema, o banco de dados e fornece uma idéia geral do sistema desenvolvido. O capítulo “Técnicas utilizadas” explica a concepção dos algoritmos utilizados e como eles funcionam. Descreve também as etapas de execução do sistema. O capítulo “Conclusões” mostra os resultados obtidos, as vantagens e desvantagens do sistema desenvolvido e sugere trabalhos futuros relacionados ao sistema.

2 - Contexto

2.1 - Introdução

"A informática surgiu para resolver os problemas que antes não existiam". Com o estado atual da informática temas que antigamente não eram tão importantes (ou até mesmo nem foram considerados) podem hoje ser discutidos, pois os computadores podem gerar soluções para alguns deles. Por exemplo, há alguns milhares de anos gerar imagens de um jogo 3D de alta resolução não era considerado um problema, mas com a existência dos computadores atuais isso é um problema.

Dentre esses novos problemas estão o estudo da inteligência artificial e da extração de informação de textos por máquinas. O objetivo principal do sistema desenvolvido nesse trabalho é a extração de informação de referências bibliográficas. Extração de informação é um problema da área de recuperação de informação. Porém esse trabalho não tem como objetivo envolver outros problemas de RI como, por exemplo, indexação e busca.

Uma das alternativas para realizar essa tarefa é utilizar a área de IA. Extração de informação de textos não é um problema que só pode ser resolvido com IA, porém IA foi a abordagem escolhida.

São tratadas a seguir as áreas de IA e de Extração de informação, com pouco foco em IA.

2.2 - Inteligência artificial

Atualmente existem dois paradigmas que são considerados os principais quando o assunte é inteligência artificial. Esses paradigmas dividem a IA em duas áreas, que são a IA simbólica e a IA conexionista.

A IA conexionista tem como inspiração o funcionamento do cérebro e a simbólica é inspirada na mente, mas não exatamente na mente como um todo, mas sim na capacidade de representação simbólica e na utilização de regras para manipular esses símbolos. Apesar da abordagem conexionista ser baseada apenas no cérebro e não tratar a mente, ela

apresenta bons resultados em vários tipos de problemas. Entre esses problemas está o reconhecimento de padrões de imagens.

Apesar do sistema desenvolvido nesse projeto utilizar regras e símbolos ele não deve ser considerado como sendo da área de IA simbólica. Isto porque apesar de possuir um conjunto de regras o sistema não é capaz de gerar novas regras de forma automática. Também não é da área de IA conexionista.

Embora não esteja classificado nas duas abordagens acima o sistema está na área de IA por ter como base o comportamento humano.

2.3 - Extração de informação

Uma das áreas da informática de grande aplicação atualmente é a área de Recuperação de Informação. Alguns dos problemas tratados nessa área são extração de informação (área tratada nesse trabalho), armazenamento da informação extraída, indexação da informação e desenvolvimento de um mecanismo de busca eficiente. Agora que foi mostrado onde a área de extração de informação se encaixa ela será tratada em mais detalhes.

Dois tipos de conjuntos de dados merecem destaque. Um desses tipos são os dados encontrados em sites. Esses dados oferecem uma facilidade extra com relação à extração de informação, que é a marcação do texto com tags HTML. Por exemplo, uma frase marcada com alguma tag de aumento de fonte informa que a frase possui maior relevância do que uma outra frase qualquer. Outra facilidade é que pode ser utilizada a freqüência de acesso ao site. Porém existe uma grande desvantagem: textos de sites geralmente não respeitam a gramática. O outro tipo de conjunto de dados é o de maior interesse nesse projeto, que são os dados na forma de texto sem formatação, ou pelo menos quase sem formatação.

A EI não é utiliza somente na área de RI. Outra área que utiliza extração de informação é a área de Data Mining (Mineração de dados), na qual muitas vezes o interesse está em se extrair informação em forma de regras. Essas regras muitas vezes permitem realizar previsões ou mostrar relacionamentos existentes entre elementos da base de dados. É muito útil quando uma empresa tem muitos dados e não sabe o que fazer com eles.

Existe um conjunto de definições relacionadas à área de EI. Uma delas é a classificação da EI em single-slot ou multi-slot. Os sistemas single-slot não relacionam campos do texto entre si. Os multi-slot conseguem fazer associações entre os campos. Por exemplo, associar os preços dos livros aos nomes dos livros.

Serão discutidas agora duas abordagens utilizadas na área de EI. Uma delas é a abordagem baseada na engenharia do conhecimento. Nessa abordagem uma pessoa gera manualmente as regras do sistema e avalia o desempenho delas, podendo modificá-las iterativamente para aumentar o desempenho do sistema. A outra abordagem é a aprendizagem automática. Nela é utilizado um conjunto de treinamento etiquetado. É utilizado algum algoritmo de aprendizagem de máquina, com o qual o sistema aprende de forma automática.

Uma das técnicas utilizadas em EI são as técnicas de PLN (Processamento de linguagem natural). Algumas das técnicas de PLN são realizar a análise sintática e semântica das palavras do texto. Outra técnica de EI é a utilização de autômatos finitos e possui muita aplicação na construção de analisadores léxicos. Uma outra forma de tratar o problema de EI é utilizar classificadores. A idéia é utilizar os classificadores em fragmentos isolados do texto. Outro tipo de técnica é a utilização de Modelos de Markov Escondidos (HMM).

Também existe a opção de se combinar várias técnicas em um único sistema, o que pode levar a um aumento de desempenho se a combinação for bem feita.

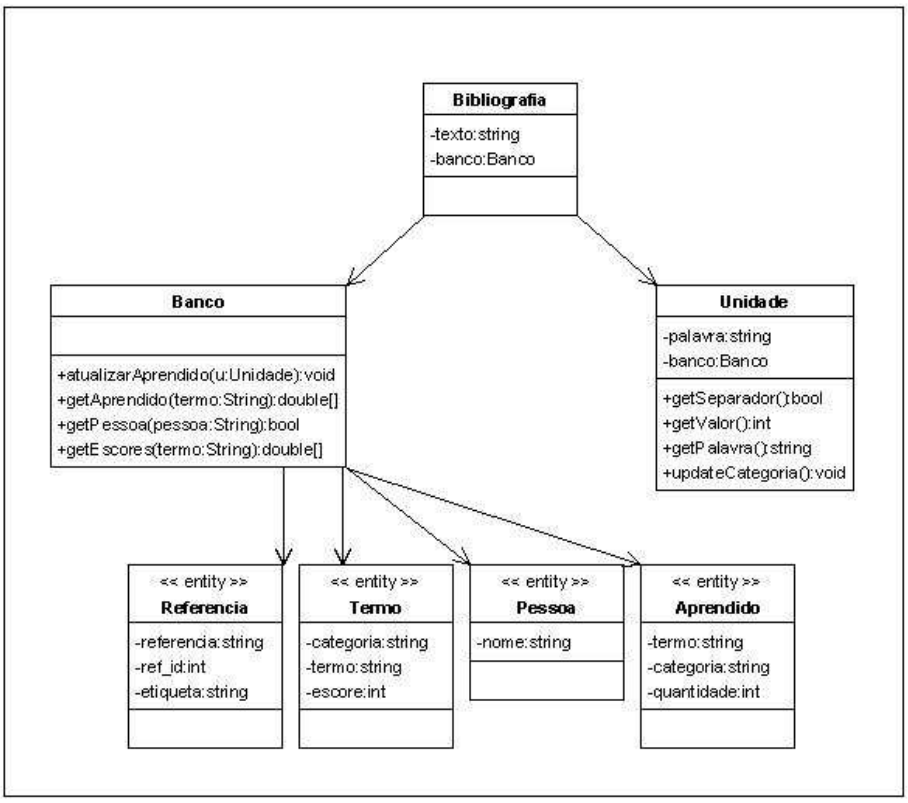
3 - Arquitetura do sistema

3.1 - Visão geral

O sistema desenvolvido processa referências bibliográficas em Português e também em Inglês, porém com foco em Português. Ele foi desenvolvido na linguagem de programação C# e faz uso de um banco de dados. Inicialmente foram desenvolvidas algumas idéias sobre como o sistema iria funcionar e sobre o que seria armazenado no banco de dados. Depois foi desenvolvida e implementada a arquitetura do sistema e foi integrada uma interface para o banco de dados. E finalmente os algoritmos foram implementados. Também foram realizadas algumas modificações na arquitetura do sistema e no banco de dados durante o desenvolvimento do sistema.

Durante a execução do sistema o conteúdo banco de dados é alterado, pois o sistema pode aprender enquanto processa uma referência. Apesar de tratar dois idiomas o foco principal e os testes estão no idioma Português.

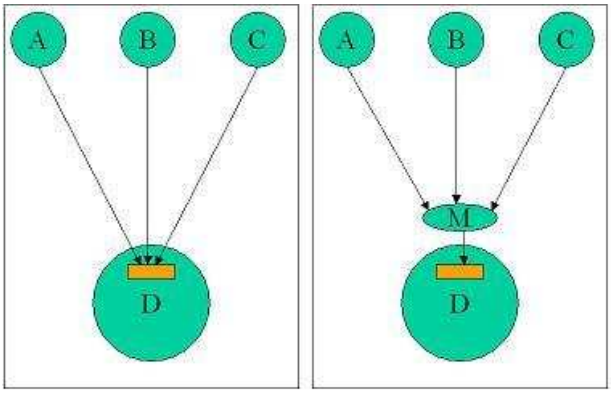
Será fornecida agora uma idéia da arquitetura do sistema.



Algumas das classes do diagrama acima podem parecer estranhas por estarem ligadas somente a classe Banco, porém elas são assim mesmo. É somente a classe banco que manipula diretamente esses objetos, que na verdade foram implementados como tabelas em um BD relacional e não chegaram a ser tratados realmente como objetos.

Outra coisa importante que deve ser considerada é que nesse diagrama de classes não estão destacados todos os métodos das classes e nem todos os atributos, apenas alguns dos principais. Esse diagrama foi feito apenas para se ter uma idéia geral da arquitetura do sistema. Isso significa que ele não descreve todos os elementos do sistema.

Os conceitos de orientação a objetos foram muito utilizados no sistema. Sua utilização permite uma melhor compreensão do sistema e reuso de código. Algo que também faz parte da orientação a objetos é a utilização de métodos, inclusive para acessar os atributos das classes, que são os conhecidos “get” e “set”. A utilização desses dois métodos foi de extrema importância. Essa importância fica clara no exemplo abaixo.



Na figura esquerda três classes modificam diretamente um atributo de D e na figura direita três classes modificam um atributo de D através de um método set. Suponha agora que a alteração desse atributo passe a necessitar da execução de algum código. No caso da figura direita só o método M precisa ser modificado, pois só ele acessa o atributo. No caso da figura esquerda as classes A, B e C precisam ser modificadas e também todas as outras classes que modifiquem esse atributo.

No sistema desenvolvido nesse trabalho um método set pode chegar a se tornar muito grande a ponto de inviabilizar o desenvolvimento caso ele não seja utilizado. Resumindo, os conceitos de orientação a objetos só foram utilizados onde havia explicitamente vantagens na utilização.

3.2 - Escopo do sistema

O escopo do sistema desenvolvido é identificar os fragmentos de referências bibliográficas e atribuir categorias a esses fragmentos. Os algoritmos do sistema não estão implementados com foco na otimização de velocidade, mas sim na otimização da classificação. A otimização da velocidade de execução não é uma tarefa diretamente relacionada à taxa de acertos na classificação do sistema e traria muito trabalho extra, principalmente porque tornaria o sistema mais complexo e conseqüentemente menos compreensível. Apesar de não estar no escopo do sistema, a otimização do código será comentada no capítulo "Conclusões", na seção "Trabalhos futuros".

Também está no escopo do sistema uma etapa de aprendizagem, armazenando novas palavras e suas respectivas categorias.

3.3 - Banco de dados

3.3.1 - Categorias

Antes de começar a explicar o banco de dados será explicado o que são e quais são as categorias consideradas no sistema. Uma categoria é uma das possíveis classes que podem ser atribuídas às partes de uma referência bibliográfica. Por exemplo, um conjunto de dados pode pertencer à categoria título. Abaixo há uma lista das categorias consideradas no sistema:

NULA: É utilizada para indicar que não se sabe qual é a categoria ou não está entre as categorias do sistema. É também a categoria inicial dos elementos de uma referência.

AUTOR: É um conjunto com um ou mais nomes e sobrenomes de autores.

TITULO: É o título da referência bibliográfica.

LOCAL: Informações sobre local como, por exemplo, cidade, estado ou universidade.

VOLUME: Como o próprio nome diz representa o volume.

SITE: Representa um endereço Web.

MES: Representa o mês.

ANO: Representa o ano.

EDITORA: Representa o nome da editora.
EDICAO: Representa o número da edição.
PAGINAS: Representa quantidades e intervalos de páginas.
PROFESSOR: Representa os nomes de professores que podem ser, por exemplo, orientadores.
ESCOLARIDADE: Representa graus de escolaridade como, por exemplo, mestrado e também a área.

O sistema não trata as possíveis fragmentações de uma categoria. Por exemplo, uma lista de autores não é quebrada em vários nomes de autores distintos.

3.3.2 - Tabelas

Agora que as categorias foram comentadas serão tratadas as principais tabelas do banco de dados. Essas tabelas contêm palavras em Português e em Inglês, sem a existência de tabelas separadas para os idiomas. Abaixo são mostradas as principais tabelas do banco de dados e para que servem:

PESSOAS: Essa tabela contém nomes próprios de pessoas que podem servir como autor ou professor. Quando o sistema encontra alguma dessas palavras em uma referência ele aumenta a probabilidade da palavra ser classificada como autor ou professor.

TERMOS: Cada registro dessa tabela possui três campos: termo,categoria e escore.
termo: Esse campo contém uma palavra.
categoria: Associa a palavra do campo “termo” a uma categoria específica.
escore: Atribui um escore para a associação acima.
Isso vai ficar mais claro com um exemplo. Considere o seguinte registro:

TERMO	CATEGORIA	ESCORE
outubro	mes	100

Quando o sistema encontra a palavra “outubro” ele associa essa palavra à categoria “mes” aumentando em 100 o escore da categoria “mes” para essa palavra. Uma palavra também pode estar associada a mais de uma categoria no banco de dados utilizando para isso mais de um registro, mas no final de todo o processamento de uma referência bibliográfica cada parte da referência só é associada a uma única categoria.

As palavras dessa tabela são armazenadas em minúsculo no banco de dados. Porém existem etapas do processamento de uma referência que tratam minúsculas e maiúsculas independentemente do banco.

APRENDIDO: Essa tabela armazena o que foi aprendido automaticamente pelo sistema. Palavras que já existam na tabela TERMOS não são acrescentadas nessa tabela. Na verdade o que é armazenado é o radical da palavra e não a palavra toda. Essa tabela possui também o campo “quantidade”, que armazena a quantidade de ocorrências do radical em cada uma das categorias.

Também existem tabelas para armazenar as referências bibliográficas, suas classificações para serem usadas nos testes e também os resultados dos testes. Essas tabelas não serão comentadas aqui porque acrescentariam apenas detalhes técnicos no sistema.

Para encontrar exemplos de registros das tabelas do sistema consulte o apêndice A.

3.3.3 - Interface

O sistema possui uma classe chamada “Banco” que é a única classe do sistema que tem acesso direto ao banco de dados. Além dessa classe possuir métodos de acesso ao banco de dados ela também faz processamento dos dados recuperados do banco. Por exemplo, existe um método que recebe uma palavra como parâmetro e retorna um array com os escores dessa palavra para cada uma das categorias existentes. Outro desses métodos é o que recebe uma palavra como parâmetro e retorna um valor booleano informando se essa palavra é um nome de pessoa.

Existem também vários métodos que são de uso interno da classe banco. Esses métodos são utilizados para auxiliar os métodos da interface. Um dos principais métodos é

o getRadical(), que recebe uma palavra e devolve algo que pode ser chamado de radical. Isso é útil no aprendizado do sistema, pois reduz o tamanho do banco de dados e ajuda o sistema a generalizar melhor. Por exemplo, as palavras “bonito” e “bonita” tem o mesmo radical e são consideradas como uma só na tabela APRENDIDO. Para mais detalhes consulte o apêndice A.

3.4 - Principais classes

Esta seção descreve as principais classes existentes no sistema. Essas classes são:

Banco: Faz a comunicação com o banco de dados. Já foi comentada na seção de banco de dados.

Bibliografia: Essa classe representa uma referência bibliográfica. Na inicialização ela recebe como parâmetros um texto, que é o próprio texto da referência bibliográfica e recebe também um objeto da classe Banco. Na etapa de testes do sistema ela recebe também a referência já classificada como parâmetro. É na classe bibliografia que ficam armazenadas as unidades. As principais etapas do processamento do texto também estão nessa classe.

Unidade: Os elementos do texto de uma referência bibliográfica são separados um a um e armazenados em objetos da classe Unidade. O conjunto desses objetos fica armazenado em um objeto da classe Bibliografia. Uma unidade pode ser, por exemplo, uma palavra ou uma vírgula.

A utilização desse conjunto de classes permite que o sistema seja mais bem compreendido. A classe Banco cuida completamente do banco de dados, permitindo que se abstraiam detalhes do banco de dados nas outras classes. Até mesmo dentro da classe Banco existe uma hierarquia na execução dos métodos. Resumindo existem métodos na classe Banco que não dependem de qual SGBD é utilizado.

A classe Bibliografia é onde está o principal em se tratando de algoritmos. É nela que está armazenado o conjunto de unidades e a maioria das regras.

3.5 - Hierarquia de execução

Esta seção mostra em qual ordem as classes têm os seus algoritmos executados. Apenas os principais métodos serão citados na seqüência abaixo. Essa seqüência é realizada em uma classificação de uma única referência.

Banco: Inicialmente é criado um objeto da classe Banco.
Bibliografia: Quando é instanciada recebe como parâmetros um texto e um objeto da classe banco, o seu texto é tratado pelo método tratarTexto(), o qual faz operações de filtragem no texto (tratar letras maiúsculas, acentos,etc) e em seguida é chamado o método gerarUnidades(), o qual transforma o texto tratado em um conjunto de "Unidade".
Unidade: Quando uma unidade é criada ela faz um processamento inicial na palavra que ela recebe. Depois disso o controle volta para o objeto Bibliografia.
Bibliografia: Realiza outra etapa do processamento através do método processar(), o qual em seguida chama o método processarB(), que realiza outra etapa.

4 - Técnicas utilizadas

4.1 - Desenvolvimento de algoritmos

Antes de começar a descrever os algoritmos do sistema é interessante comentar sobre algo muito ligado a algoritmos e à programação, que é a famosa frase “Reinventar a roda”. O que ela significa para você? Significa fazer trabalho desnecessário? Se a resposta é sim, talvez você deva tentar se lembrar o que significa a frase “Tentar melhorar algo”.

Albert Einstein definiu insanidade como “Continuing to do the same things and expecting different results”. Em português é “fazer sempre as mesmas coisas e esperar resultados diferentes”.

Devemos desconsiderar a idéia de que reinventar a roda é sempre algo ruim. Também não dá pra dizer que é sempre bom. Deve haver um equilíbrio entre reinventar e reutilizar.

Devemos também considerar uma frase: “Como saber a melhor forma de se atingir um objetivo sem antes saber qual é o objetivo?”. Explicando melhor, qual é o melhor caminho do ponto A ao ponto B? Você talvez pense: É o caminho que leva menos tempo, menos trabalho, etc. Com isso surge a pergunta: Menos tempo para o quê? Não foi dito que o objetivo era chegar no menor tempo ao ponto B. O objetivo poderia ser, por exemplo, conhecer alguns dos caminhos de A até B, servindo talvez de base para chegar futuramente ao ponto C.

4.2 - Surgimento das idéias

A primeira idéia a surgir foi a de utilizar regras de classificação para categorizar os fragmentos de uma referência bibliográfica. A próxima etapa foi perceber a necessidade de se utilizar um banco de dados para armazenar algumas palavras e símbolos, pois alguns símbolos já têm um significado quase que predefinido como, por exemplo, a palavra “Fevereiro”. Também foi decidido que os fragmentos deveriam ter escores associados a cada categoria. Depois disso foi definida uma versão inicial do banco de dados com algumas tabelas.

Agora que a seqüência de idéias foi mostrada de forma resumida o surgimento delas será explicado melhor. Um dos problemas percebidos foi a escolha das regras a serem utilizadas. Sabendo-se que humanos realizam muito bem está tarefa de classificação, podem ser utilizadas algumas das regras que os humanos usam para classificar. Por isso a geração de regras foi manual. Foi definido um conjunto de regras inicial, o qual foi testado e melhorado iterativamente. Os humanos também têm facilidade na extração de informação por terem conhecimento prévio do significado de várias palavras. Então foi definido que o sistema deveria armazenar previamente o significado de alguns termos. Porém esse significado não é sempre o mesmo, podendo variar de acordo com o contexto.

A idéia do escore funciona de forma semelhante a uma taxa de probabilidade apesar de não ser um número entre 0 e 1 e não ser realmente a probabilidade. O mais importante é que esse escore serve para que o sistema possa fazer medições e comparações, pois da forma como o sistema foi desenvolvido não dá para se calcular todas as probabilidades reais.

Os algoritmos utilizados estão agrupados em cinco etapas de processamento do sistema. Uma grande parte do código está relacionada às regras de classificação, as quais foram desenvolvidas e aperfeiçoadas iterativamente. Após o desenvolvimento de uma versão inicial do sistema foi decidido que o sistema deveria aprender. Com isso foi criada a tabela APRENDIDO no banco de dados. Porém o sistema não aprende novas regras, mas sim novos símbolos. As próximas seções desse capítulo descrevem as principais etapas do funcionamento do sistema. A descrição fornecida é de forma resumida.

4.3 - Etapa 1: pré-processamento

Quando um objeto Bibliografia é instanciado ele inicia um pré-processamento para filtrar o texto. O processo é iniciado por uma separação dos símbolos existentes no texto. O sistema separa palavras, vírgulas, ponto, etc. Depois disso o texto pré-processado é utilizado na geração das unidades. Quando cada uma das unidades é instanciada ela recebe um texto, que é um dos símbolos da referência. Esse texto pode ser, por exemplo, uma palavra ou uma vírgula.

4.4 - Etapa 2: classificação inicial

As unidades processam o texto recebido da seguinte forma:

- É verificado se o símbolo é um separador como, por exemplo, vírgula ou dois pontos. Essa informação é armazenada;
- Se a palavra tem exatamente duas letras e ambas são maiúsculas e não são um algarismo romano então ela tem um aumento no escore de local por ter possibilidade de ser uma sigla de algum estado;
- Depois são processadas outras regras para verificar se é uma sigla de local, um nome, uma inicial ou se a palavra possui hífen.
- Os escores da unidade são inicialmente zerados, com exceção da categoria NULA que tem um pequeno valor inicial positivo;
- Os escores são atualizados de acordo com a tabela TERMOS e com a tabela APRENDIDO;
- È verificada a possibilidade da unidade representar um ano;
- A terminação da palavra é considera. Por exemplo, terminações “ing” e “mente” aumentam o escore da unidade para título;
- È verificada a possibilidade da unidade representar a categoria página.

Esclarecendo melhor a utilização da tabela APRENDIDO, considere que estão armazenados em cada registro um radical (R), uma categoria (C) e a quantidade de ocorrências (Q) relacionada ao radical e à categoria. Se a quantidade total de ocorrências de um radical é menor do que 2, então esse radical ainda não é utilizado na classificação. Considere agora que um radical tem 10 ocorrências e 7 dessas ocorrências pertence à categoria AUTOR. O calculo do aumento do escore da categoria AUTOR é feito da seguinte forma:
((7 / 10) * 100) - 50. Caso dê negativo o valor passa a ser zero. O mesmo é feito para as outras categorias.

4.5 - Etapa 3: classificação

Depois de gerar as unidades o sistema passa para a etapa de classificação. Nessa etapa é processada uma grande quantidade de regras. As regras modificam os escores das unidades ao invés de simplesmente fornecerem uma classificação absoluta e inflexível para elas. Essas regras são implementadas com if, else, switch, etc. A ordem de execução das regras influencia o resultado. Para ter uma idéia de como são essas regras consulte o apêndice B.

4.6 - Etapa 4: filtragem

Após a classificação das unidades o sistema filtra as classificações das unidades. Uma dessas filtragens é verificar se alguma das categorias aparece em mais de um grupo da referência. Para isso considere como grupos as maiores seqüências de unidades de uma mesma categoria. Caso encontre mais de um grupo de uma mesma categoria o sistema analisa os grupos, podendo em alguns casos mudar a categoria de algum grupo para nula em todas as suas unidades. O sistema também procura por unidades de categoria nula e classifica essas unidades de acordo com as categorias que ainda não foram encontradas na referência. Resumindo, o sistema tenta corrigir os próprios erros.

4.7 - Etapa 5: aprendizagem

Terminada a classificação o sistema armazena novas palavras no banco de dados e atualiza palavras existentes no banco. Isso tudo é feito na tabela APRENDIDO. A tabela TERMOS não é modificada pelo sistema e a tabela APRENDIDO não recebe como entrada palavras que já existam na tabela TERMOS.

Na tabela APRENDIDO são armazenados os radicais das palavras aprendidas, as categorias e a quantidade de ocorrências. A idéia de armazenar o radical das palavras e não a palavra toda permite que o sistema utilize melhor o que foi aprendido, pois o surgimento de radicais semelhantes ocorre com mais freqüência do que o de palavras semelhantes.

5 - Conclusões

5.1 - Resultados

5.1.1 - Medidas utilizadas

Alguns tipos de símbolos foram ignorados nos testes. Alguns desses símbolos são: vírgula, ponto final, hífen, etc. Isso significa que eles não entram na contagem de símbolos para os testes.

Foram realizados testes com e sem aprendizagem automática. Nos testes com aprendizagem o que é aprendido em uma referência é utilizado em todas as referências seguintes. Abaixo estão as medidas utilizadas nos testes de desempenho do sistema:

taxa de acerto: Indica o percentual de classificações realizadas corretamente considerando todos os elementos classificáveis de todas as referências de teste.

precisão e cobertura: São considerados todos os símbolos classificáveis de todas as referências de teste para essas duas medidas.

5.1.2 - Testes

Os testes foram realizados com um conjunto de 50 referências bibliográficas da UFPE, todas em Português. Não há um conjunto de treinamento, pois existem apenas regras, um pequeno conjunto de termos pré-definidos, e um pequeno conjunto com nomes de pessoas. Esse conjunto de nomes não foi retirado de referências bibliográficas e contém apenas 87 nomes. A tabela de termos contém apenas 123 registros.

A etapa de aprendizagem também pode ser realizada durante os testes. Quando uma referência é classificada pelo sistema essa classificação é utilizada na aprendizagem. Porém a aprendizagem não utiliza as etiquetas (classificação pré-definida dos termos de uma referência), pois se utilizasse as etiquetas para aprender a etapa deixaria de ser teste a passaria a ser treinamento. O que é aprendido em uma referência é utilizado nas referências seguintes.

As categorias existentes no conjunto de testes são: AUTOR, TITULO, LOCAL, ANO, ESCOLARIDADE e PROFESSOR.

Exemplo de uma referência de teste:

TAVARES, Dione Jorge de Souza. Esplenose associada a esplenectomia e ligadura de veia gástrica esquerda em crianças esquistossomóticas. Análise seqüencial das imunoglobulinas G, A e M dos componentes 3 e 4 do sistema complemento.Recife: UFPE, 1992. Mestrado em Cirurgia. Orientador: Prof. Carlos Teixeira Brandt

Abaixo estão as tabelas de desempenho do sistema:

Desempenho com a aprendizagem desativada:

Categoria	Precisão	Cobertura
AUTOR	84,44%	90,47%
TITULO	85,12%	87,78%
LOCAL	88,28%	98,00%
ANO	100,00%	100,00%
ESCOLARIDADE	100,00%	100,00%
PROFESSOR	95,78%	98,14%

Taxa de acerto: 92,66%

Apesar de não conter um conjunto de treinamento o sistema obteve um desempenho elevado. Esse desempenho foi obtido usando as regras do sistema, alguns poucos termos (210 registros) do banco de dados e a capacidade do sistema de auto-correção.

Desempenho após uma seqüência de testes com aprendizagem e uma sem aprendizagem:

Categoria	Precisão	Cobertura
AUTOR	88,78%	94,28%
TITULO	88,30%	90,05%
LOCAL	93,33%	98,00%
ANO	100,00%	100,00%
ESCOLARIDADE	100,00%	100,00%
PROFESSOR	95,78%	98,14%

Taxa de acerto: 94,20%

A aprendizagem automática de novos símbolos melhorou o desempenho do sistema. Algumas taxas aumentaram e algumas se mantiveram as mesmas. Porém nenhuma diminuiu.

Desempenho após duas seqüências de testes com aprendizagem e uma sem aprendizagem:

Categoria	Precisão	Cobertura
AUTOR	93,75%	100,00%
TITULO	90,22%	90,48%
LOCAL	93,33%	98,00%
ANO	100,00%	100,00%
ESCOLARIDADE	100,00%	100,00%
PROFESSOR	100,00%	100,00%

Taxa de acerto: 95,55%

Após aprender com o mesmo conjunto de teste o sistema continuou a aumentar as taxas de desempenho e novamente nenhuma taxa diminuiu.

Desempenho após três seqüências de testes com aprendizagem e uma sem aprendizagem:

Categoria	Precisão	Cobertura
AUTOR	93,75%	100,00%
TITULO	90,22%	90,48%
LOCAL	93,33%	98,00%
ANO	100,00%	100,00%
ESCOLARIDADE	100,00%	100,00%
PROFESSOR	100,00%	100,00%

Taxa de acerto: 95,55%

Como pode ser observado, o sistema manteve as mesmas taxas.

5.1.3 - Vantagens e desvantagens

Esta seção discute algumas vantagens e algumas desvantagens do sistema desenvolvido.

Vantagens:
1 - Não é necessário um conjunto de treinamento;
2 - O sistema aprende automaticamente o significado de novos símbolos;
3 - As regras manuais podem chegar a ser muito bem elaboradas, pois são originadas de humanos, os quais executam muito bem a tarefa de extração de informação;
4 - Algumas regras podem ser reaproveitadas em outros sistemas, mesmo que os sistemas não sejam do mesmo domínio;
5 - O armazenamento de radicais ao invés da palavra toda permite que o sistema generalize melhor, podendo utilizar melhor o conhecimento adquirido;
6 - Como pôde ser visto, o sistema obteve um bom desempenho;
7 - O sistema possui uma etapa para corrigir os próprios erros.

Desvantagens:
1 - As regras devem ser geradas manualmente;
2 - O especialista pode apresentar dificuldades para gerar as regras;
3 - O sistema não explica a lógica utilizada para uma classificação;
4 - O sistema não armazena o significado de um conjunto de símbolos, armazenando apenas símbolos individuais.

5.2 - Trabalhos futuros

5.2.1 - Como otimizar

A otimização do código-fonte do sistema para otimizar a velocidade do processamento deixa o sistema muito mais complexo. Isso aumentaria a complexidade da arquitetura, tornando o sistema menos compreensível e portanto mais propenso a falhas por parte do programador. O desenvolvimento do sistema também seria mais demorado.

Ao observar que o sistema faz utilização intensa de switch, percebe-se que é possível substituir alguns desses switches por arrays. Como sabemos arrays podem ter os seus elementos acessados de forma direta. Algumas seqüências de ifs podem ser substituídas por alguma estrutura que permita uma execução mais rápida como, por exemplo, uma árvore de decisão. O banco de dados também poderia ser substituído por algo desenvolvido especialmente para esse sistema.

Para ficar mais clara a utilização de arrays em otimização será dado um exemplo. Imagine que um algoritmo precise verificar em qual conjunto estão as últimas três letras de uma palavra (desconsiderando acentos nas letras e outras coisas). Utilizando um longo if ou um longo switch o código poderia ficar lento. Poderíamos criar uma função que trate cada letra como um conjunto de 5 bits (permite todas as letras do nosso alfabeto) e une essas 3 letras (15bits) em 2 bytes, o que é suficiente. Por utilizar 15bits todas as terminações possíveis poderiam ser armazenadas em um array de 32768 posições, o qual seria utilizado com acesso direto. Dependendo do percentual desse array que o maior conjunto representa, a utilização desse array pode ser considerada como boa.

Alguém poderia dizer: “Por que não utilizar um banco de dados?”. A resposta para essa pergunta seria: “Quem disse que um banco de dados não pode ser utilizado? As informações podem inicialmente ser lidas de um banco de dados e armazenadas em um array, o qual fica em memória RAM, que é de acesso mais rápido do que o disco rígido.”.

Conclui-se que o trabalho extra de otimização seria muito complexo, tornando-se uma boa opção caso haja algum interessado na utilização do sistema.

5.2.2 - Melhorias no banco de dados

Uma das melhorias que poderia ser feita no banco de dados seria acrescentar tabelas de informações no banco como, por exemplo, uma tabela com nomes de lugares. Também poderiam existir tabelas que armazenassem as classes gramaticais das palavras e as relações entre elas, considerando as categorias.

Outra melhoria relacionada ao banco de dados seria carregar parte do banco de dados na memória RAM para ter um acesso mais rápido.

5.2.3 - Aprendizagem melhorada

Uma forma de melhorar a aprendizagem seria não aprender quando o sistema perceber uma alta probabilidade de ter classificado errado. Uma das formas de se fazer isso é diminuindo um escore de confiança quando o sistema encontra elementos de mesma categoria desagrupados. Mas isso somente se o tipo de texto tratado não contém mais de um grupo de uma mesma categoria.

Apêndice A: Registros das tabelas

São mostrados abaixo exemplos de alguns registros de algumas tabelas existentes no banco de dados do sistema.

Tabela TERMOS:

TERMO	CATEGORIA	ESCORE
agosto	mes	100
as	titulo	70
de	titulo	15
mestrado	escolaridade	100
pags	paginas	100
v	volume	60

Tabela APRENDIDO:

Observação: Essa tabela é inicialmente vazia. O sistema preenche automaticamente.

TERMO	CATEGORIA	QUANTIDADE
almeid	autor	3
andrad	autor	3
arrud	autor	2
recif	local	50
ufp	local	48
acid	titulo	3
analis	titulo	2
aspect	titulo	3

Apêndice B: Exemplos de código-fonte

Os exemplos desse apêndice servem para que se tenha uma idéia do código-fonte do sistema.

Exemplo de um case para uma unidade da categoria ano:

```
case CAT_ANO:
     if(this.unidades[i-2].getCategoria() == CAT_ANO
     && this.unidades[i-2].getValor() > 0)
     {
          this.incEscore((i-2),CAT_ANO,-120.0);
          this.incEscore(i,CAT_ANO,-120.0);
     }
     if(this.unidades[i-1].getCategoria() == CAT_ANO
     && this.unidades[i-1].getValor() > 0)
     {
          this.incEscore((i-1),CAT_ANO,-120.0);
          this.incEscore(i,CAT_ANO,-120.0);
     }
          this.incEscore((i-1),CAT_ANO,-35.0);
          this.incEscore((i-2),CAT_ANO,-25.0);

     if(this.unidades[i-1].getValor() > 0 && this.unidades[i-
     1].getValor() <= 12)
     {
          this.incEscore((i-1),CAT_MES,45.0);

          this.incEscore(i,CAT_ANO,30.0);
     }
     if(this.unidades[i-2].getValor() > 0 && this.unidades[i-
     2].getValor() <= 12)
     {
          this.incEscore((i-2),CAT_MES,35.0);
     }
     if(this.unidades[i+1].getSeparador() == false)
     {
          this.incEscore((i),CAT_ANO,-150.0);
     }
     break;
```

Exemplo de código para mudar a categoria de uma unidade que está próxima a unidades da categoria “titulo”:

```
//palavra no meio do titulo vira titulo
for(i=1;i < (this.totalUni-1);i++)
{
     if(this.unidades[i-1].getSeparador() == false
     && this.unidades[i-1].getCategoria() == CAT_TITULO)
     {
          this.unidades[i].incEscore(this.unidades[i].getCate
          goria(),-30.0);
          this.unidades[i].incEscore(CAT_TITULO,this.unidades
          [i-1].getEscore());
          this.unidades[i].updateCategoria();
     }
     if(this.unidades[i].getCategoria() == CAT_TITULO
     && this.unidades[i+1].getCategoria() == CAT_TITULO
     && this.unidades[i-1].getSeparador() == false)
     {
          this.unidades[i-
          1].incEscore(this.unidades[i].getCategoria(),-
          30.0);
          this.unidades[i-
          1].incEscore(CAT_TITULO,this.unidades[i].getEscore(
          ));
          this.unidades[i-1].updateCategoria();

     }
}
```

Referências bibliográficas

[1] Eduardo Fraga do Amaral e Silva; “UM SISTEMA PARA EXTRAÇÃO DE INFORMAÇÃO EM REFERÊNCIAS BIBLIOGRÁFICAS BASEADO EM APRENDIZAGEM DE MÁQUINA” (dissertação de mestrado)

[2] John E. Hopcroft, Jeffrey D. Ullman e Rajeev Motwani - “Introdução à Teoria deAutômatos. Linguagens e Computação”, editora: Campus Ltda. - 2003

[3] Ricardo Baeza e Berthier Ribeiro - “Modern Information Retrieval”, editora: ACM Press 1999

[4] Stuart J. Russell e Peter Norvig - “Artificial Intelligence - A Modern Approach”, editora: Prentice Hall, 2002

[5] Germano Crispim Vasconcelos; “Fundamentos de inteligência artificial. Abordagem conexionista (Redes neurais)” (PDF)

[6] Olinda Nogueira Paes Cardoso; “Recuperação de Informação” (PDF)

Assinaturas

13 de Fevereiro de 2006

_____

Flávia de Almeida Barros

(Orientadora)

_____

Max José Lins Timóteo

(Aluno)