DESENVOLVIMENTO DE FERRAMENTAS PARA A CONSTRUÇÃO ÁGIL DE EXTRATORES DE INFORMAÇÃO

Matheus Silva Santos[1]
Cassiano Bueno Silva[2]
Eraldo R. Fernandes[3]

[1]Instituto Federal de Goiás/Campus Jataí/Técnico em Informática, mths.ss@msn.com
[2]Instituto Federal de Goiás/Campus Jataí/Técnico em Informática, cassiano929@hotmail.com
[3]Instituto Federal de Goiás/Campus Jataí, eraldo.fernandes@ifg.edu.br

Resumo

A Internet é o maior repositório de informação de toda a história e grande parte das informações disponíveis nesta imensa rede estão em formato textual. Textos escritos em linguagem natural (Português, Inglês, etc.) são adequados para o ser humano. Entretanto, as informações contidas em um texto são difíceis de serem processadas por um programa de computador. A área de processamento de linguagem natural tem o objetivo de criar algoritmos que consumam textos e convertam as informações ali contidas para um formato adequado ao processamento computacional. Isto é, um formato estruturado, como uma tabela ou um banco de dados, por exemplo. Atualmente, a área de aprendizado de máquina fornece as técnicas mais eficazes para o processamento de linguagem natural. Algoritmos baseados em aprendizado de máquina são capazes de derivar um modelo de linguagem natural a partir de exemplos fornecidos como entrada. A aplicação adequada destes algoritmos exige diversas ferramentas de apoio para obtenção, anotação e gerenciamento dos exemplos necessários. Neste trabalho, desenvolvemos um protótipo de um sistema computacional para assistir a anotação de textos e geração de exemplos para algoritmos de aprendizado de máquina. Futuramente, este sistema integrará estratégias de aprendizado ativo e transferência de domínio para agilizar o processo de anotação.

Palavras-chave: Extração de Informação; Aprendizado de Máquina; Processamento de Linguagem Natural; Anotação de Texto

INTRODUÇÃO

A Internet é o maior repositório de informação de toda a história. Somente a versão em inglês da *Wikipedia*, por exemplo, é aproximadamente 50 vezes maior do que a *Encyclopædia Britannica*, a maior enciclopédia impressa em língua inglesa [Wikipedia, 2012]. Grande parte da Internet é composta por textos escritos em linguagem natural, ou seja, textos escritos em inglês, francês, português, ou algum outro idioma. A quantidade de informação presente em todo este conteúdo textual é tão grande que as aplicações mais usadas atualmente na Internet são os sistemas buscadores como, por exemplo, o Google. Estes sistemas facilitam a busca por informações, filtrando o imenso volume de documentos de acordo com termos de interesse fornecidos pelo usuário. Entretanto, sistemas buscadores não compreendem totalmente as informações contidas em um documento.

Na Figura 1, apresentamos um texto que contém algumas informações sobre o jogador de futebol Kaká e outras entidades relacionadas a ele. Um sistema buscador moderno representa este texto simplesmente como um conjunto de palavras. No máximo, estes sistemas são capazes de reconhecer que, por exemplo, as palavras Real e Madrid são altamente relacionadas e,

portanto, compõem um único termo, ou seja, Real Madrid. Entretanto, estes sistemas não são capazes de reconhecer as relações entre as entidades mencionadas no texto. Por exemplo, o fato de que Kaká nasceu na cidade de Brasília não é reconhecido como uma relação específica entre o jogador Kaká e a cidade de Brasília.

Kaká nasceu em **Brasília** no dia **22 de abril de 1982** e é um jogador de futebol brasileiro. Atualmente, ele atua pelo **Real Madrid**. Este clube tem sede em **Madri** que é a capital da **Espanha**.

Figura 1: texto com informações sobre algumas entidades. As entidades mencionadas estão em negrito.

Técnicas de processamento de linguagem natural (PLN) [Manning e Schütze, 1999] permitem a extração de informações contidas em documentos, convertendo estas informações para um formato estruturado. Existem diversas maneiras para representar informações de maneira estruturada. O modelo relacional [Codd, 1970], por exemplo, é uma das maneiras mais conhecidas e nós utilizaremos este modelo apenas para ilustrar o processo de extração de informação. Na Figura 2, apresentamos um exemplo de como o PLN pode ser aplicado para extrair informações de um texto. Neste exemplo, as informações presentes no texto são reconhecidas e convertidas para um formato relacional. Por exemplo, a informação de que o jogador Kaká nasceu em Brasília no dia 22/04/1982 é identificada no texto e transferida para um formato relacional, mais especificamente, para um registro da relação Nascimento. Uma relação pode ser representada como uma tabela, como é ilustrado na figura. Outra informação presente no texto é que Kaká é jogador de futebol e atua no clube Real Madrid. Esta informação pode ser representada como um registro da relação Emprego. Outras informações podem ser extraídas deste pequeno fragmento de documento e representadas neste formato estruturado.

Kaká nasceu em Brasília no dia 22 de abril de 1982 e é um jogador de futebol brasileiro. Atualmente, ele atua pelo Real Madrid. Este clube tem sede em Madri que é a capital da Espanha.

PLN

Relação Nascimento

Pessoa	Cidade	Data
Kaká	Brasília	22/04/1982

Relação Emprego

Pessoa	Profissão	Empresa
Kaká	Jogador de Futebol	Real Madrid

Figura 2: Extração de informação usando PLN.

Dada a quantidade de documentos presentes na Internet, estas ferramentas de PLN podem ser usadas para manter atualizada uma grande base de conhecimento com informações sobre praticamente qualquer domínio de interesse. No caso do futebol, por exemplo, algumas relações de interesse são:

- *Jogador* – informações pessoais de jogadores: nome, data de nascimento, salário, valor de mercado, clubes onde atuou, etc.

- *Equipe* – informações sobre times de futebol: nome, data de fundação, presidente, etc.
- *Gol* – informações sobre cada gol marcado em toda a história do futebol: marcador, goleiro que sofreu o gol, partida na qual o gol foi marcado, qual posição do campo o marcador estava, etc.

Uma vez que estas informações estão estruturadas, programas de computador podem acessá-las de maneira simples e eficiente, possibilitando diversas aplicações. Dado esta base de conhecimento, um sistema de busca pode responder perguntas altamente complexas como, por exemplo, *Qual jogador do Flamengo de todos os tempos marcou mais gols contra em partidas realizadas no Maracanã?* Mantendo-se uma base de conhecimento sobre política, pode-se monitorar ações de políticos, notícias, opiniões de jornalistas e até da opinião pública, monitorando-se as redes sociais, por exemplo. É possível também monitorar tudo que é publicado na Internet sobre uma empresa, uma marca ou um produto. Tudo isto pode ser realizado de forma automática, em tempo real, através de ferramentas de PLN.

Os algoritmos de PLN mais eficazes são baseados em técnicas de aprendizado de máquina (AM) [Mitchell, 1997]. AM consiste em técnicas de análise de padrão a partir de um conjunto de exemplos, derivando-se modelos que, posteriormente, são capazes de identificar informações de interesse em novos exemplos desconhecidos previamente. Existem diversos paradigmas de aprendizado de máquina e o mais estudado é chamado aprendizado supervisionado. Neste caso, um exemplo é composto pelos dados de entrada juntamente com a saída esperada. Por exemplo, para derivar um identificador de nomes de pessoa, um exemplo corresponde a uma frase e a correta identificação dos nomes de pessoas que são mencionadas na frase. O volume de dados necessário para se derivar um bom modelo através de um algoritmo de aprendizado supervisionado é, em geral, muito grande. Por isto, a utilização destes algoritmos implica em um alto custo alto para a criação dos exemplos anotados com a resposta desejada. Um ou mais especialistas precisam despender horas de trabalho para anotar nomes de pessoas em uma grande quantidade de texto.

Para superar este problema, algumas técnicas podem ser aplicadas para minimizar o custo de anotação. Aprendizado ativo [Settles, 2012] consiste em usar algoritmos de AM específicos para auxiliar o especialista durante o processo de anotação. Basicamente, estes algoritmos de aprendizado escolhem os exemplos mais informativos para que o especialista os anote, ao invés de anotar todos os exemplos. Escolhendo os exemplos que fornecem mais informação sobre o fenômeno de interesse, menos exemplos são anotados para se obter um bom extrator. Outra técnica muito utilizada para se minimizar o esforço de anotação é a transferência de domínio [Pan e Yang, 2010]. Neste caso, a ideia básica consiste em utilizar dados anotados em um domínio, para derivar um modelo a ser aplicado em outro domínio. Por exemplo, ao se desenvolver um extrator de nomes de jogador de futebol, pode-se utilizar dados anotados com nomes de pessoas em geral e, desta forma, minimizar a quantidade de exemplos a serem anotados com nomes de jogador de futebol especificamente.

Existem diversas propostas na literatura de métodos para aprendizado ativo e transferência de domínio [Settles, 2012; Pan e Yang, 2010]. Cada qual com suas vantagens e desvantagens, suas potencialidades e limitações, dependendo de diversos fatores relacionados ao domínio onde estes métodos são aplicados. Portanto, ao se aplicar algum destes métodos, é necessário analisar seu impacto na aplicação envolvida e, também, os custos envolvidos. Além disto, a aplicação adequada destes métodos, principalmente em ambiente de produção, exige algumas ferramentas e também infra-estrutura específica, principalmente, no que se diz respeito a sistemas computacionais. Os métodos de aprendizado ativo para PLN envolvem a anotação de

informações complexas sobre um documento. Por isto, o uso de interfaces ricas é muito importante para a adequada visualização e edição destas informações. Outro ponto essencial na aplicação destas técnicas é o gerenciamento dos modelos de extração que devem ser criados e atualizados constantemente pelos algoritmos de aprendizado ativo e transferência de domínio.

Neste trabalho, apresentamos o protótipo de um sistema computacional para a anotação de *tokens* (palavras e sinais de pontuação) em um texto, que é um item básico para a maioria dos sistemas de PLN. Futuramente, este protótipo será estendido para um sistema completo de desenvolvimento ágil de extratores de informação.

VISÃO GERAL DO PROTÓTIPO

O protótipo desenvolvido é baseado no caso de uso de anotações de tokens. Neste caso de uso, existem quatro atores principais: (i) o usuário anotador (*Annotator*); (ii) a interface do sistema com o anotador (*Frontend*); (iii) o módulo de negócios(*Backend*); e (iv) o sistema de gerenciamento de banco de dados (*DBMS*). Na Figura 3, apresentamos o diagrama de sequência típico deste caso de uso, ilustrando as principais interações entre os atores envolvidos.

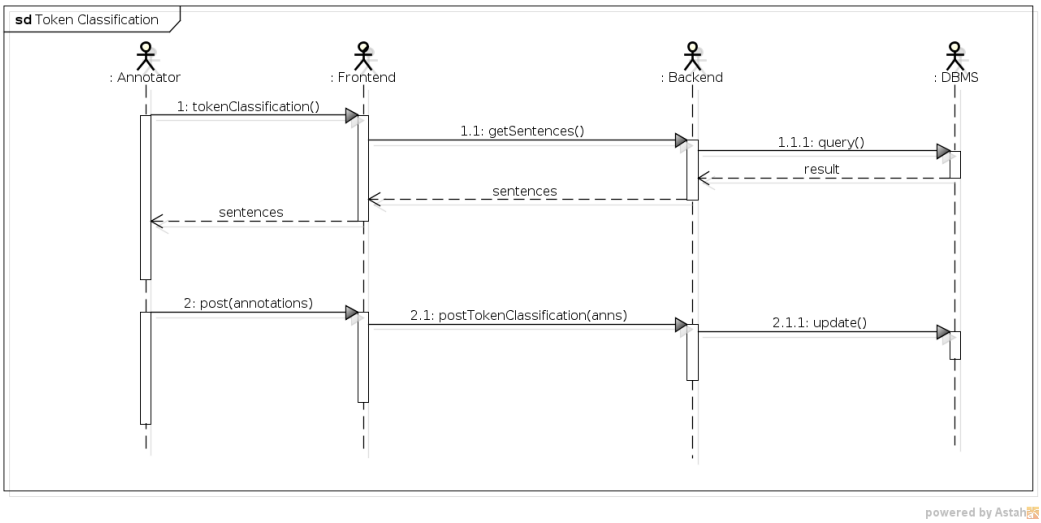

Figura 3: Diagrama de Caso de Uso - Anotação de Tokens

O *Annotator* inicia o caso de uso selecionando a opção *tokenClassification* disponível no *Frontend*. O *Frontend* solicita ao *Backend* uma lista de frases a serem anotadas (revisadas e corrigidas). O *Backend* consulta o *DBMS* para obter as frases que necessitam de anotação naquele momento e retorna estas frases ao *Frontend*. Por sua vez, o *Frontend* exibe ao *Annotator* as frases com as anotações disponíveis, permitindo que o *Annotator* corrija as anotações e/ou forneça novas anotações.

INTERFACE DO ANOTADOR – *FRONTEND*

Na Figura 4, apresentamos a visualização que o *Annotator* tem do *Frontend*. Neste caso, para simplificar a ilustração, utilizamos apenas uma frase simples. Em uma interação real, o *Annotator* terá acesso a uma lista de frases.

Acima de cada token da frase, é apresentado uma etiqueta que, opcionalmente, pode estar em branco. O *Annotator* pode então selecionar alguma etiqueta para ser alterada. Uma vez

concluída a revisão das frases apresentadas, o *Annotador* envia as informações revisadas ao sistema que, por sua vez, atualiza o *DBMS* e retorna novas frases a serem anotadas.

A principal tecnologia utilizada no desenvolvimento do *Frontend* é o framework para desenvolvimento web *Django*. Este sistema é baseado na linguagem Python e oferece uma ótima alternativa para o desenvolvimento de sistemas web de maneira rápida. A interface em si é representada utilizando-se HTML e CSS, que são as duas principais tecnologias utilizadas para apresentação de sistemas web.

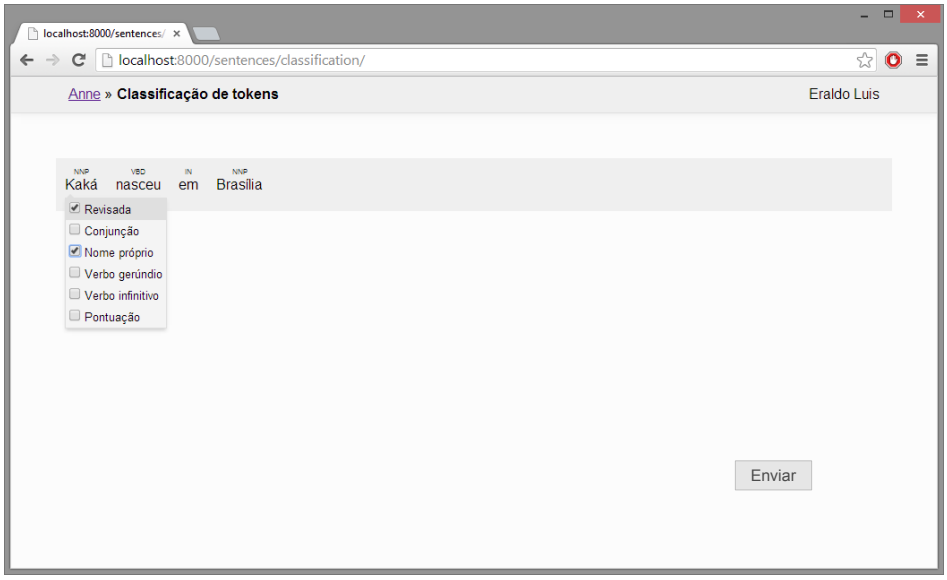

Figura 4: Frontend do protótipo

MÓDULO DE NEGÓCIOS – *BACKEND*

O cerne do sistema é o *Backend*, que permite a interação do anotador com os dados a serem anotados e, futuramente, gerenciará os módulos de Aprendizado Ativo e Transferência de Domínio. O *Backend* segue o formato de um *web service*, um sistema web que provê serviços para aplicações clientes (atualmente, o único cliente é o *Frontend*) utilizando uma interface bem definida. Nossa implementação de *Backend* é baseada na linguagem *Java*, mais especificamente no sistema *Apache Tomcat*.

A interface do *Backend* com seus clientes é baseada em JSON, uma linguagem de representação de dados vastamente utilizada em sistemas web, devido a sua simplicidade e eficiência. Desta forma, a inclusão de novos clientes torna-se bastante simples.

A interação do *Backend* com o *DBMS* é realizada através do conceito de Mapeamento Objeto-Relacional (ORM, da sigla em inglês) utilizando o sistema *Hibernate* que implementa a *Java Persistence API* (JPA). ORM é atualmente o padrão de representação de dados em bases de dados persistentes. *Hibernate* é o sistema ORM mais utilizado para a linguagem Java.

SISTEMA GERENCIADOR DE BASE DE DADOS – *DBMS*

Utilizamos o sistema *PostgreSQL*, um DBMS robusto e de código fonte aberto que possui ótima compatibilidade com Hibernate.

RESULTADOS ALCANÇADOS

A anotação de tokens é uma atividade essencial em diversos sistemas de PLN. Portanto, o protótipo desenvolvido será de grande utilidade para o projeto de pesquisa em questão. Além disso, a arquitetura do protótipo contempla muitos aspectos de outras funcionalidades que serão implementadas futuramente.

Além dos recursos desenvolvidos, este projeto também trouxe inúmeros benefícios para os estudantes envolvidos. Eles puderam trabalhar com tecnologias extremamente relevantes no mercado de trabalho e também no meio acadêmico.

REFERÊNCIAS

E. F. Codd. A Relational Model of Data for Large Shared Data Banks. Em: **Communications of the ACM**, 1970.

M. Collins. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. Em: **Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing**, p. 1-8, 2002.

E. R. Fernandes e U. Brefeld. Learning from partially annotated sequences. Em: **Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)**, Athens, Greece, 2011.

C. Freitas, P. Rocha e E. Bick. Floresta Sintá(c)tica: Bigger, thicker and easier. Em: Antonio Teixeira, Vera Lucia Strube de Lima, Lus Caldas de Oliveira, and Paulo Quaresma, editors, Computational Processing of the Portuguese Language, volume 5190 of **Lecture Notes in Computer Science**, p. 216-219, 2008.C. D. Manning e H. Schütze. Foundations of Statistical Natural Language Processing, MIT Press (1999).

T. Mitchell. **Machine Learning**, McGraw Hill (1997).

S. J. Pan e Qi. Yang. A Survey on Transfer Learning. Em: **IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE)**. Volume 22, No. 10, Pages 1345-1359, October 2010.

F. Rosenblatt. **The Perceptron – a perceiving and recognizing automaton**. Report 85-460-1, Cornell Aeronautical Laboratory (1957).

B. Settles. Active Learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, June 2012, Vol. 6, No. 1, p. 1-114. Morgan & Claypool Publishers.

Wikipedia. **Size comparisons**. End.: http://en.wikipedia.org/wiki/Wikipedia:Size_comparisons. Acessado em: 01/11/2012.