4
Arquitetura

Neste trabalho estamos interessados apenas em mostrar como pode ser construída uma camada de apoio a moderação para sistemas de comentários. Sendo assim, vamos tomar como base o sistema de comentários utilizado na Globo.com para explicar alguns conceitos sobre sistemas de comentários, sabendo que os conceitos básicos utilizados aqui permitem que este trabalho seja aplicado em qualquer outro sistema do tipo.

O sistema de comentários da Globo.com é responsável pelos comentários em diversos sites, tanto da própria Globo.com quanto da TV Globo. Para evitar que os comentários sobre economia sejam moderados por um moderador de esportes ou da novela, cada tópico (ou assunto) comentado é classificado em categorias de acordo com a área em que está inserido. São categorias existentes: g1-economia, g1-educação, globoesporte-flamengo, globoesporte-vôlei entre outros. Embora o propósito desta classificação tenha sido criado para controle de acesso, indiretamente ela ajuda a definir um tema, ou **contexto**, agrupando os comentários de assuntos comuns em uma mesma categoria. Em geral, os comentários dentro da mesma categoria possuem alguns critérios comuns para aprovação ou reprovação. Esta classificação dos assuntos, de acordo com o contexto, será útil ao final do trabalho para construção de classificadores específicos ao contexto.

Para entender conceitualmente a relação entre as entidades no sistema de comentários, veja o diagrama da figura 1. Nele podemos observar que temos diversos comentários sobre um Tópico e que este é classificado em uma categoria.

Figura 1 – Relacionamento entre as entidades do sistema de comentários

4.1.Visão geral da arquitetura

Com o intuito de facilitar a manutenção e evolução do ambiente e permitir a produção de componentes de reuso, o sistema de moderação foi dividido em camadas, cada um responsável por uma parte do sistema.

Para conseguir auxiliar o moderador na classificação, foram utilizados algoritmos de aprendizado de máquina, o que levou a construção de uma camada para este propósito, com o nome de *aprendizagem*. Esta camada, é ainda subdividida em duas: *classificadores*, onde está a implementação dos algoritmos de classificação de comentários e a camada *extratores*, onde está a implementação dos métodos de extração de atributos.

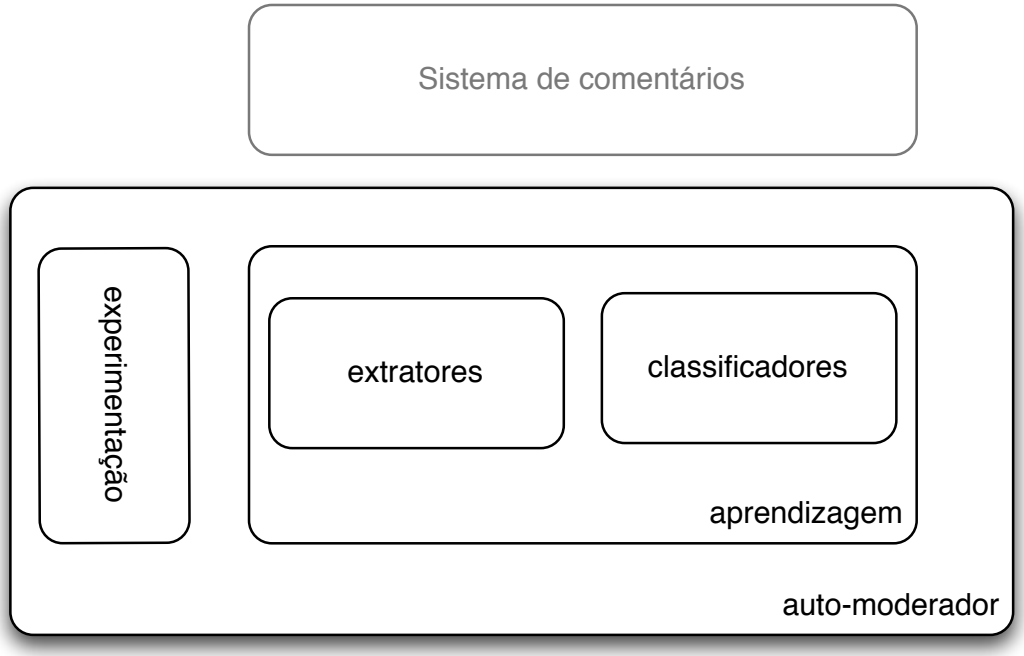

Figura 2 - Arquitetura do sistema

Com a camada de aprendizagem isolada, é possível reutilizá-la para fins de experimentação, com corpus e parâmetros variados, e também para integração com a ferramenta de comentários. Para experimentação uma API de linha de comando foi construída, permitindo combinar todos os extratores com os algoritmos de classificação. O apêndice C detalha o uso desta API. Para integração com o sistema de comentários, basta este enviar os comentários classificados manualmente para o treino do moderador e para cada comentário novo consultar a predição.

Na figura 2 o diagrama de classes da camada de aprendizagem é apresentado. Como este trabalho foi implementado em linguagem Python, algumas considerações serão feitas:

- Python não possui atributos privados, mas possui um mecanismo para evitar conflito de atributos entre classes na herança. Os atributos com esta característica devem iniciar com __, mas não podem terminar com tal. Embora isto não seja estritamente um atributo privado, foi tratado aqui como tal, pois é o mecanismo mais próximo do conceito oferecido pela linguagem. Apesar da necessidade do atributo iniciar com duplo sublinhado, no diagrama de classes o duplo sublinhado não está presente pois é subentendido pela marcação de privado.
- Python não possui classes abstratas nem interface. Para indicar que uma classe é abstrata ou interface foi incluído no nome da classe o sufixo I. Observe que isto é apenas uma convenção deste trabalho.

Nas seções abaixo as camadas serão detalhas, mostrando seus componentes internos e as classes que a implementam.

4.2.Camada de Aprendizagem

É a principal camada, responsável por abrigar os algoritmos de moderação, que inclui os classificadores e os métodos de extração de atributos. O diagrama da figura 3 mostra as classes existentes e a relação entre elas.

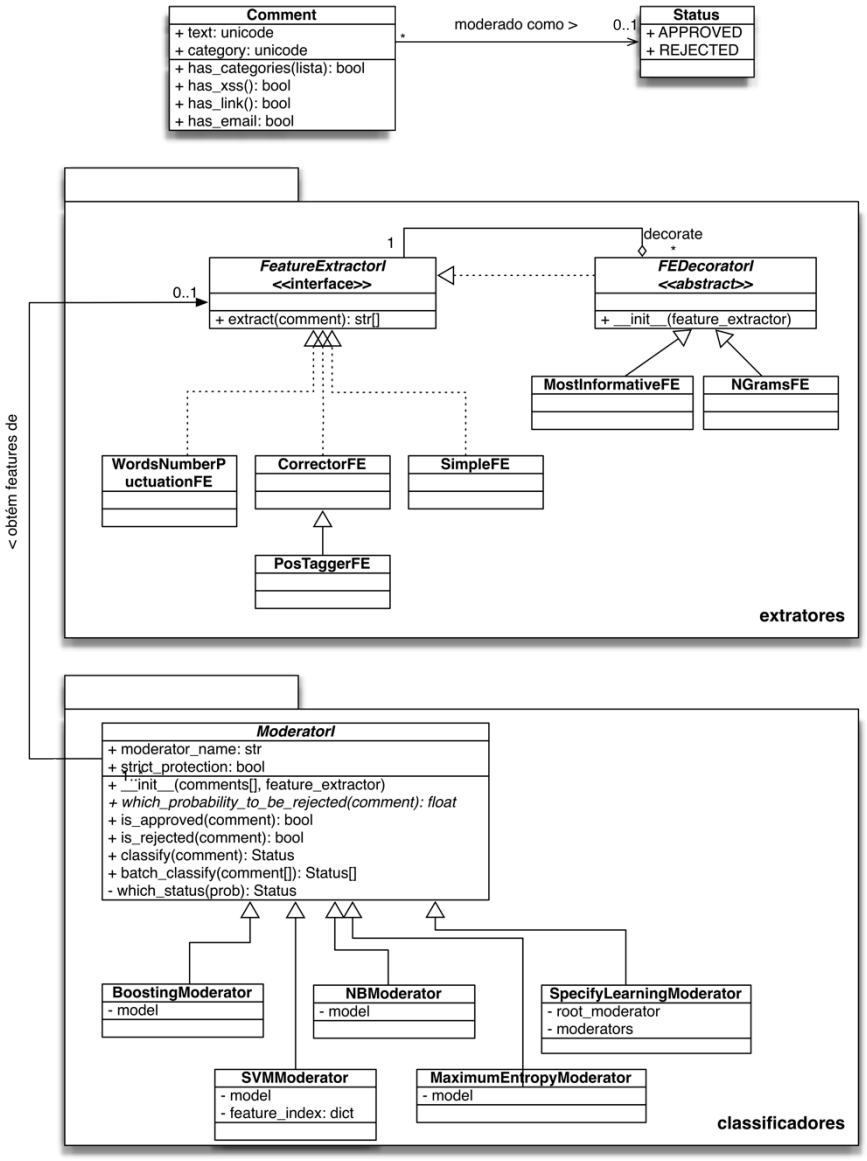

Figura 3 - Diagrama de classes da camada aprendizagem

A classe *Comment* é responsável por representar um comentário. Quando ele está classificado, o atributo status indica sua classe, que pode ser aprovado (*APPROVED*) ou rejeitado (*REJECTED*).

As classes responsáveis por realizar a extração dos atributos de um comentário herdam de *FeatureExtractorI* e devem implementar um método

chamado *extract* responsável por retornar os atributos a partir de um comentário.

Abaixo segue a descrição de cada subclasse de *FeatureExtractorI*. Os detalhes sobre os métodos são descritos na seção 3.2.

- *SimpleFE*: implementa o método de extração descrito na seção 3.2.1, porém considerando como atributos somente palavras e atributos.
- *WordsNumberPuctuationFE*: análogo ao SimpleFE, mas também considera como atributo sinais de pontuação.
- *CorrectorFE*: Análogo ao *SimpleFE*. Porém corrige a grafia das palavras automaticamente, utilizando para isto o corretor ortográfico *Aspell* (ATKINSON, 2004). As regras para correção ortográfica estão descritas na seção 3.2.2
- *PosTaggerFE*: Este método utiliza as palavras corrigidas pelo *CorrectorFE* e adiciona a elas a classificação gramatical. O método utilizado para classificar gramaticalmente o texto está descrito na seção 3.2.3
- *MostInformativeFE*: Este *decorator* (GAMMA, HELM, *et al.*, 1994) é responsável por utilizar somente os atributos mais informativos.
- *NGramsFE*: Este *decorator* é responsável por combinar os atributos da classe decorada em subconjuntos de n-grams. A seção 3.2.4 detalha o processo.

Já as classes responsáveis por realizar o aprendizado e a classificação de comentários têm como classe pai a classe *ModeratorI*, que implementa o padrão de projeto *Template method* (GAMMA, HELM, *et al.*, 1994). Os únicos métodos que precisam ser implementados pelas subclasses são *__init__*, que é o construtor e deverá realizar o treino do algoritmo e o método *which_probability_to_be_rejected*, que deve retornar um número entre 0 e 1.0 com a probabilidade de um comentário ser rejeitado. A probabilidade do comentário ser aprovado é complementar a rejeição.

A classe *ModeratorI* possui ainda um atributo que merece destaque: *strict_protection*. Quando este atributo contém o valor verdadeiro, os comentários que possuem *tags html,* códigos *javascripts*, links ou e-mails são diretamente rejeitos, sem necessidade de consultar o algoritmo de moderação, porém eles ainda são utilizados para treino.

Abaixo a descrição das subclasses de *ModeratorI*. Os detalhes da implementação dos algoritmos são descritos no Capítulo 3.

- *BoostingModerator*: Utiliza o algoritmo *Boostexter* para moderação.
- *SVMModerator*: Utiliza o algoritmo *SVM* para moderação.
- *NBModerator*: Utiliza o algoritmo *Naive Bayes* para moderação.
- *SpecificLearningModerator*: Um *decorator* que utiliza os classificadores especificados acima para moderar, construindo modelos específicos a cada categoria do comentário e um modelo global. Os detalhes sobre o funcionamento deste moderador está descrito na Seção 5.4.2.1.