Estudo de técnicas de detecção de plágio em código fonte

Autor: Luiz Henrique Dias Lima
Orientador: André Ricardo Backes

Sumário

1. Introdução

- O plágio consiste em entregar algo que não é de sua autoria sem os devidos créditos;
- Diversas formas de obter códigos fontes;
- Diferentes formas e níveis de “ocultar” o plágio;
- Dificuldade em identificar possíveis dependências manualmente.

1. Introdução

- Busca-se automatizar essa tarefa;
- Deseja-se apenas resultados relevantes;
- Principais dificuldades:
  - Definição do limiar *T*;
  - Definição das classes de equivalência;
  - Nuances de cada atividade.

2. Objetivos

- Objetivo geral: Estudar e combinar técnicas de detecção de plágio em códigos fontes já existentes, visando resultados melhores do que os vistos atualmente.
- Objetivos específicos:
  - Desenvolver um algoritmo capaz de gerar *streams* de *tokens;*
  - Investigar o método mais adequado para ponderar os termos;
  - Investigar a eficácia do modelo vetorial e coeficiente de Dice.
- Hipótese: O uso de técnicas de RI poderá trazer resultados satisfatórios.

3. Metodologia

- Três *softwares* utilizados
  - JPlag, *structure measure;*
  - MOSS, *structure measure;*
  - YouPlag, *attribute measure.*

3. Metodologia

- Base de dados: 3 bases contendo problemas de níveis distintos
  - Base 1: Operações em *skiplist;*
    - Nível: Intermediário;
    - Quantidade: 14 códigos.
  - Base 2: Operações em árvore B;
    - Nível: Avançado;
    - Quantidade: 15 códigos.
  - Base 3: Operações matemáticas;
    - Nível: Iniciante
    - Quantidade: 12 códigos.
- Foi criada uma lista com os possíveis plágios de cada base;
- Eficácia calculada através da precisão e exatidão;

4. Desenvolvimento

- Tokenização
  - Definição das classes de equivalência;
  - Uso de expressões regulares (REGEX);
  - Geração dos *4-grams.*

Código 1: Código original retirado da base.

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #define MAXN 5
4.
5. int geraNivel(){
6.         int lvl = 1+rand()%MAXN;
7.         return lvl;
8. }
```

Tokens:
YFWADOÇVCRZ

4-grams:
YFWA FWAD WADO ADOÇ DOÇV OÇVC ÇVCR VCRZ

Figura 5 – Representação de *4-grams* de um trecho de código.

Fonte: O autor

4. Desenvolvimento

- Índice Invertido
  - Armazenado em memória através do uso de uma tabela *hash;*
  - Utilizado posteriormente para ponderar os termos.
- Ponderação de termos
  - Duas formas de cálculo:
    - TF-IDF:

$$\text{tf-idf}_{i,j} = \begin{cases} f_{ij} \times idf(t_i), & \text{se } \mathrm{f}_{ij} \geq 1 \\ 0, & \text{caso contrário} \end{cases}$$

    - WF-IDF:

$$\text{wf-idf}_{i,j} = \begin{cases} (1 + \log f_{ij}) \times idf(t_i), & \text{se } \mathrm{f}_{ij} \geq 1 \\ 0, & \text{caso contrário} \end{cases}$$

4. Desenvolvimento

- Consulta
  - Similaridade por cosseno:

$$\cos(D_i, Q) = \frac{\sum_{j=1}^{t} d_{ij} \times q_j}{\sqrt{\sum_{j=1}^{t} d_{ij}^2 \times \sum_{j=1}^{t} q_{ij}^2}}$$

  - Coeficiente de Dice:

$$dice(D1, D2) = 2\frac{D1 \times D2}{\sum_{j=1}^{t} D_{1j}^2 \times \sum_{j=1}^{t} D_{2j}^2}$$

4. Desenvolvimento

- Limiarização
  - Objetivo: Separar possíveis cópias de não cópias;
  - Dificuldade em definir o *T* de similaridade inicial;
  - Limiar de Otsu.

4. Desenvolvimento

- Apenas resultados superiores ao limiar *T* são mostrados.

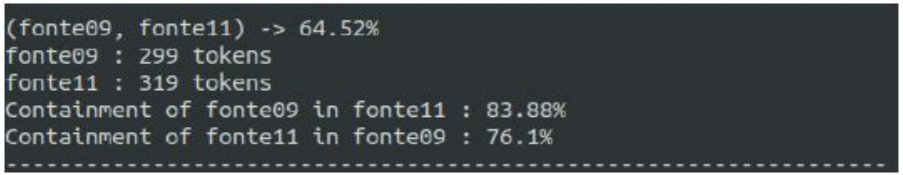

Figura 6 – Nível de similaridade entre um par de documentos.

Fonte: O autor

5. Resultados - Limiar de 50% - Base 1

- JPlag e MOSS apresentaram resultados de precisão inferiores;
- Uso do TF-IDF influenciou a **precisão**;
- O WF-IDF influenciou negativamente na **exatidão**

Tabela 3 – Resultados na base de exercícios de *skiplist*.

Software	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
JPlag	7	3	4	0	43%	100%
MOSS	8	3	5	0	38%	100%
YouPlag -dice -tfidf	5	3	2	0	60%	100%
YouPlag -cosine -tfidf	5	3	2	0	60%	100%
YouPlag -cosine -wfidf	2	2	0	1	100%	67%
YouPlag -dice -wfidf	2	2	0	1	100%	67%

5. Resultados - Limiar de 50% - Base 2

- Exatidão perfeita para todas as execuções;
- Aparição de um falso negativo no esquema de ponderação TF-IDF;
- Diferentes esquemas de ponderação influenciaram de maneiras distintas os resultados.

Tabela 4 – Resultados na base de exercícios de árvore B.

Software	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
JPlag	1	1	0	0	100%	100%
MOSS	1	1	0	0	100%	100%
YouPlag -dice -tfidf	2	1	1	0	50%	100%
YouPlag -cosine -tfidf	2	1	1	0	50%	100%
YouPlag -cosine -wfidf	1	1	0	0	100%	100%
YouPlag -dice -wfidf	1	1	0	0	100%	100%

5. Resultados - Limiar de 50% - Base 3

- Precisão de 100%, mas uma baixa exatidão de 33% foi observada;
- Menos trabalho manual de inspeção dos códigos, mas com um custo altíssimo;
- Necessidade de automatizar a obtenção do limiar *T*.

Tabela 5 – Resultados na base de exercícios de operações matemáticas.

Software	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
JPlag	1	1	0	2	100%	33%
MOSS	1	1	0	2	100%	33%
YouPlag -dice -tfidf	1	1	0	2	100%	33%
YouPlag -cosine -tfidf	1	1	0	2	100%	33%
YouPlag -cosine -wfidf	1	1	0	2	100%	33%
YouPlag -dice -wfidf	1	1	0	2	100%	33%

5. Resultados - Limiar de Otsu - Base 1

- Exatidão desejável;
- Queda acentuada nos valores de precisão;
- Necessidade de avaliar muitos códigos manualmente.

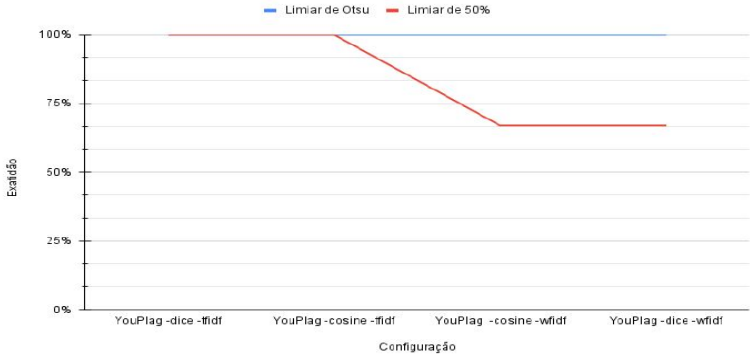

Figura 10 – Comparação entre os valores de exatidão.

Fonte: O autor

Tabela 6 – Resultados na base de exercícios de *skiplist* utilizando o limiar de Otsu.

Software	Limiar%	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
YouPlag -dice -tfidf	20	14	3	11	0	21%	100%
YouPlag -cosine -tfidf	20	14	3	11	0	21%	100%
YouPlag -cosine -wfidf	20	9	3	6	0	33%	100%
YouPlag -dice -wfidf	20	9	3	6	0	33%	100%

5. Resultados - Limiar de Otsu - Base 2

- Observou-se outra queda nos valores de precisão;
- Valores de exatidão se mantiveram.

Tabela 7 – Resultados na base de exercícios de árvore B utilizando o limiar de Otsu.

Software	Limiar%	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
YouPlag -dice -tfidf	20	10	1	9	0	10%	100%
YouPlag -cosine -tfidf	20	10	1	9	0	10%	100%
YouPlag -cosine -wfidf	20	4	1	3	0	25%	100%
YouPlag -dice -wfidf	20	4	1	3	0	25%	100%

5. Resultados - Limiar de Otsu - Base 3

- Melhora significativa nos valores de exatidão;
- Queda na precisão com o uso do WF-IDF;
- Proporcionalidade entre precisão e exatidão.

Tabela 8 – Resultados na base de exercícios de operações matemáticas utilizando o limiar de Otsu.

Software	Limiar%	Documentos Recuperados	Documentos Relevantes	Falsos Positivos	Falsos Negativos	Precisão	Exatidão
YouPlag -dice -tfidf	30	2	2	0	1	100%	67%
YouPlag -cosine -tfidf	30	2	2	0	1	100%	67%
YouPlag -cosine -wfidf	10	5	2	3	1	40%	67%
YouPlag -dice -wfidf	10	5	2	3	1	40%	67%

6. Conclusão

- As técnicas de RI implementadas proveram bons resultados na presença do limiar de 50%;
- TF-IDF mostrou-se mais eficaz nas bases apresentadas;
- A característica do IDF trouxe alguns benefícios;
- O limiar de Otsu não gerou valores de precisão satisfatórios na maioria das execuções. Ao passo que a exatidão melhorou ou se manteve.

6. Trabalhos futuros

- Investigar formas de obter o limiar eficazmente;
  - Elementos visuais (dados estatísticos, grafos de dependência entre os códigos, etc.);
  - Automação através de técnicas de IA.

Referências

Whale, G. Identification of program similarity in large populations. The computer

journal, v. 33, n. 2, p. 140–146, 1990.

Manning, C. D.; Raghavan, P.; Schütze , H. An Introduction to Information Retrieval.

[S.l.]: Cambridge University Press, 2009

Prechelt, L.; Malpohl, G.; Philippsen, M. Finding plagiarisms among a set of programs

with jplag. Journal of Universal Computer Science, v. 8, n. 11, p. 1015–1038, 2003.

Obrigado!