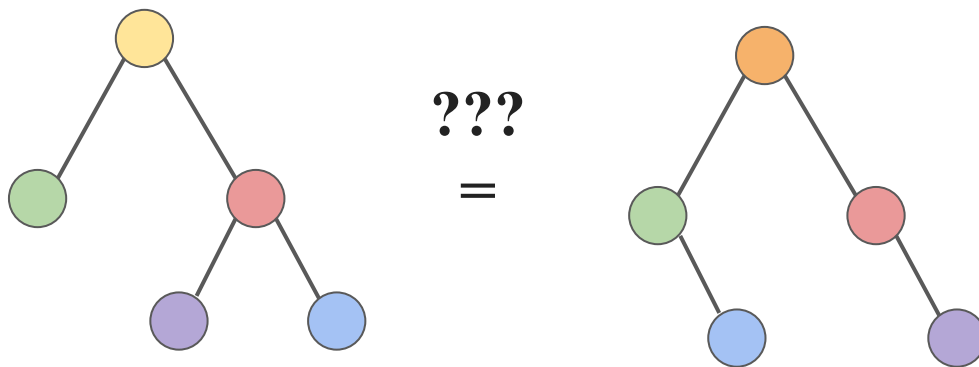


Revisiting Code Similarity Evaluation with Abstract Syntax Tree Edit Distance

Yewei Song, Cedric Lothritz, Daniel Tang, Tegawendé F. Bissyandé and Jacques Klein
[2024]



ProblematicS

```
class Atome {
protected:
    char s;
    int z;
    int n;

public:
    // Constructeur vide
    Atome() : s(' '), z(0), n(0) {}

    // Constructeur à paramètres
    Atome(char s, int z, int n) : s(s), p(z), n(n) {}

    // Procédure pour afficher le symbole chimique de l'atome
    void Affiche() const;
    //Fonction pour afficher le symbole chimique de l'atome
    char Element() const;
    //Fonction pour retourner la masse de l'atome
    int Masse() const;
};

//Procédure qui affiche le symbole chimique de l'atome
void Atome::Affiche() const {
    cout << s;
}

//Fonction qui affiche le symbole chimique de l'atome
char Atome::Element() const {
    return s;
}

//Fonction qui retourne la masse de l'atome
int Atome::Masse() const {
    return z + n;
}
```

```
// Déclaration de la classe Atome
class Atome {
private:
    char symbole; // Symbole chimique
    int z; // Nombre de protons
    int n; // Nombre de neutrons

public:
    // Constructeur vide
    Atome() : symbole('\0'), z(0), n(0) {}

    // Constructeur avec paramètres
    Atome(char s, int proton, int neutron) : symbole(s), z(proton), n(neutron) {}

    // Affichage du symbole chimique
    void Affiche() const {
        std::cout << symbole;
    }

    // Retourne le symbole chimique
    char Element() const {
        return symbole;
    }

    // Calcul de la masse (somme des protons et neutrons)
    int Masse() const {
        return z + n;
    }
};
```

```
//Classe Atome
class Atome
{
private:
    char symbol;
    int N=0;
    int Z=0;

public:
    //constructeur vide
    Atome();
    Atome(char a,int b,int c);
    void Affiche() const;
    //const pour les fonction qui d'après votre cours ne me modifie pas moi même :)
    char Element() const ;
    int Masse() const ;
};

Atome::Atome()
{}

Atome::Atome(char a,int b,int c)
{
    symbol=a;
    Z=b;
    N=c;
}

void Atome::Affiche() const
{
    cout<<symbol;
}

char Atome::Element() const
{
    return symbol;
}

int Atome::Masse() const
{
    return N+Z;
}
```



ProblematicS

```
class Atome {  
protected:  
    int s;
```

```
// Déclaration de la classe Atome  
class Atome {  
private:  
    int symbole; // Symbole chimique  
    int z;
```

```
//Classe Atome  
class Atome  
{  
private:  
    int s;
```

EMERGENCY
MEETING



GPT's bad habits

```
class Atome {
protected:
    char s;
    int z;
    int n;

public:
    // Constructeur vide
    Atome() : s(' '), z(0), n(0) {}

    // Constructeur à paramètres
    Atome(char s, int z, int n) : s(s), p(z), n(n) {}

    // Procédure pour afficher le symbole chimique de l'atome
    void Affiche() const;
    //Fonction pour afficher le symbole chimique de l'atome
    char Element() const;
    //Fonction pour retourner la masse de l'atome
    int Masse() const;
};

//Procédure qui affiche le symbole chimique de l'atome
void Atome::Affiche() const {
    cout << s;
}

//Fonction qui affiche le symbole chimique de l'atome
char Atome::Element() const {
    return s;
}

//Fonction qui retourne la masse de l'atome
int Atome::Masse() const {
    return z + n;
}
```



```
// Déclaration de la classe Atome
class Atome {
private:
    char symbole; // Symbole chimique
    int z; // Nombre de protons
    int n; // Nombre de neutrons

public:
    // Constructeur vide
    Atome() : symbole('\0'), z(0), n(0) {}

    // Constructeur avec paramètres
    Atome(char s, int proton, int neutron) : symbole(s), z(proton), n(neutron) {}

    // Affichage du symbole chimique
    void Affiche() const {
        std::cout << symbole;
    }

    // Retourne le symbole chimique
    char Element() const {
        return symbole;
    }

    // Calcul de la masse (somme des protons et neutrons)
    int Masse() const {
        return z + n;
    }
};
```



```
//Classe Atome
class Atome
{
private:
    char symbol;
    int N=0;
    int Z=0;

public:
    //constructeur vide
    Atome();
    Atome(char a,int b,int c);
    void Affiche() const;
    //const pour les fonction qui d'après votre cours ne me modifie pas moi même :)
    char Element() const ;
    int Masse() const ;
};

Atome::Atome()
{}

Atome::Atome(char a,int b,int c)
{
    symbol=a;
    Z=b;
    N=c;
}

void Atome::Affiche() const
{
    cout<<symbol;
}

char Atome::Element() const
{
    return symbol;
}

int Atome::Masse() const
{
    return N+Z;
}
```

Chatty

Clean code

“High” level programming

Hard to use specific libraries

How to compare code ?

1 - Whitespace insensitivity

not affected by text style

easy

2 - Noise Suppression

no false positives with small matches

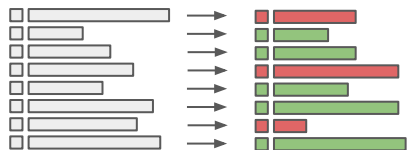
medium

3 - Position Independence

resistant to permutations

hard

Text Diff / Overlapping



$$\text{Score} = \frac{\text{Number of matches}}{\text{Total number of lines}}$$

strings to fingerprint
SCAM

Winnowing



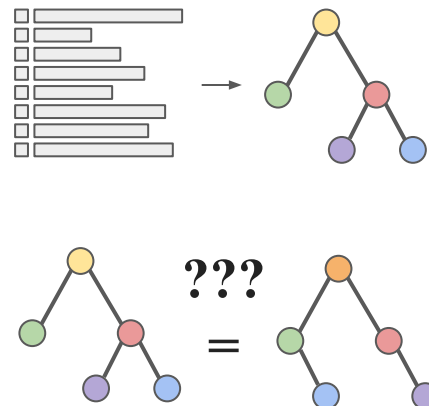
Hashes: {1245, 596, 102, 1513, ...}

Windows: {{1245, 596}, {596, 102}, ...}

Fingerprints : {596, 102, 91, 1, ...}

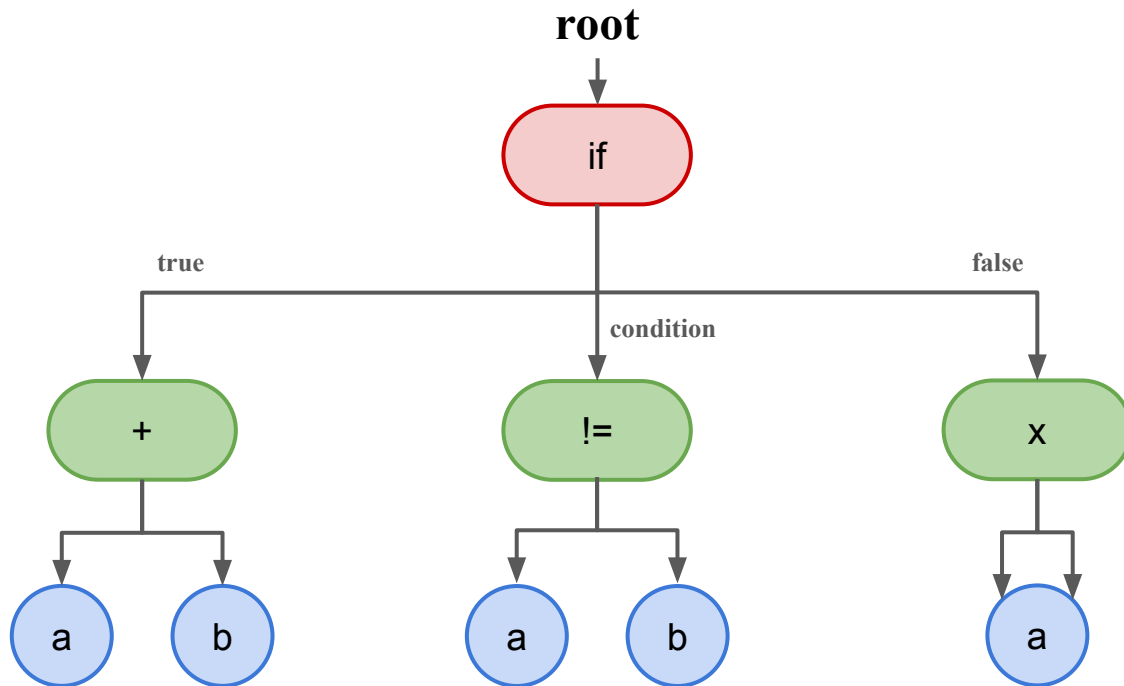
Moss

Syntax Tree

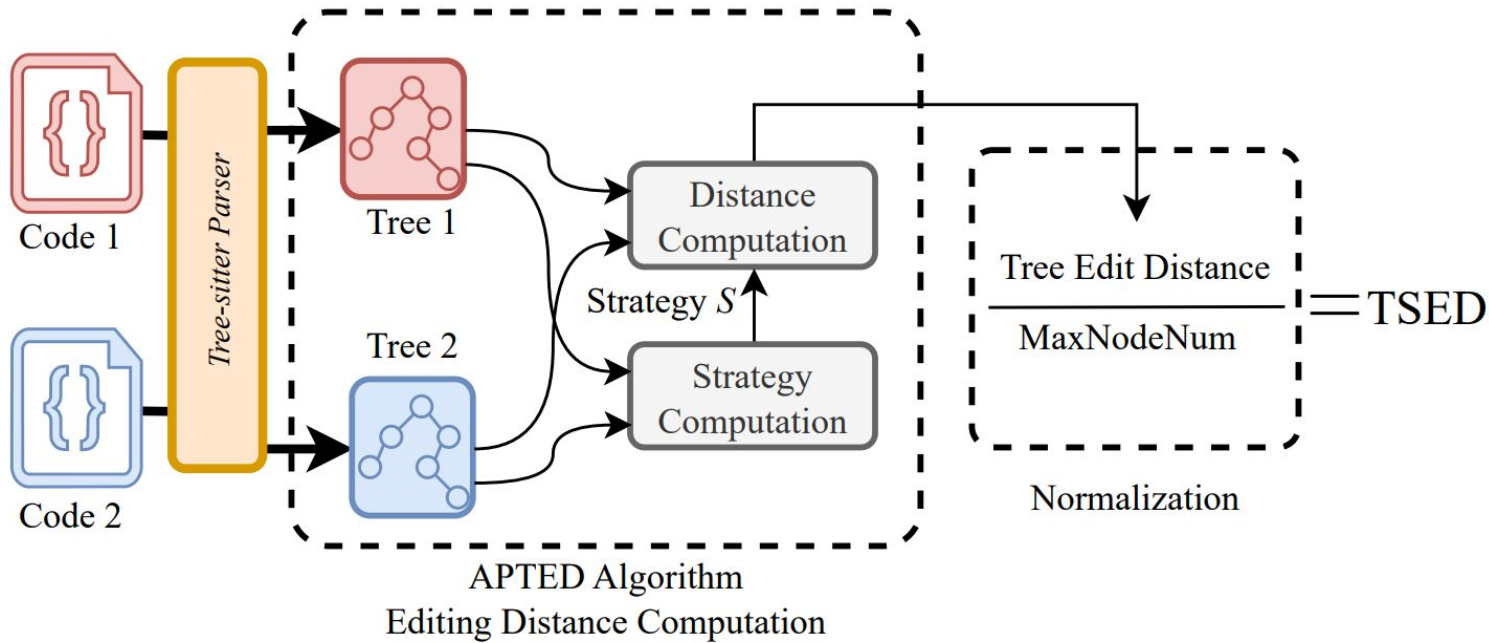


Abstract Syntax Tree (AST)

$a + b$ **if** $a \neq b$ **else** a^2

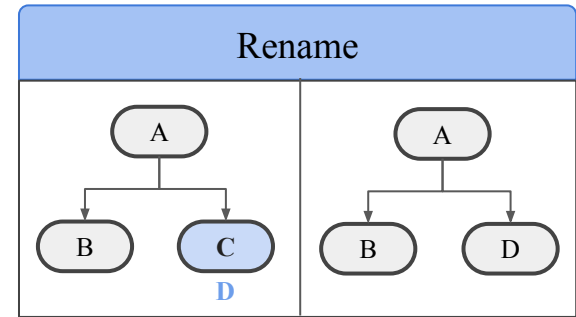
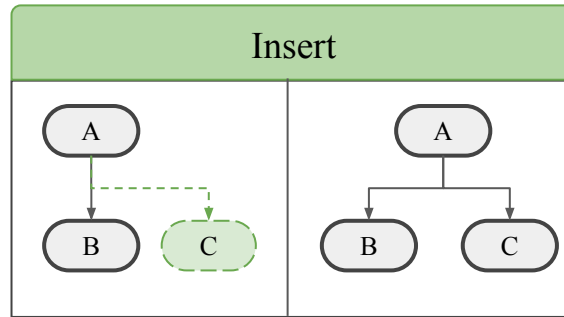
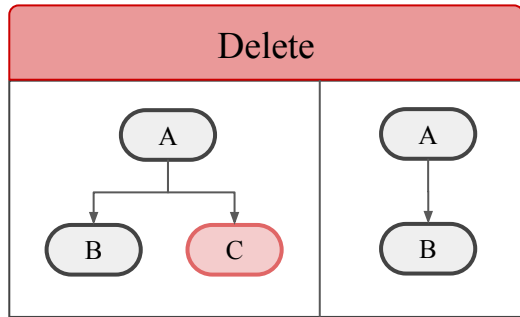


Pipeline



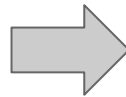
Tree Distance Measure

Operations



Tree Similarity of Edit Distance

$$\Delta(G_1, G_2) = \min_{ops} \sum_{i=1}^n w(op_i)$$



$$TSED = \max\left\{1 - \frac{\delta}{MaxNodes(G_1, G_2)}, 0\right\}$$

Min #operation needed to go from G1 to G2

Normalisation [0,1]

Op Weights : w_{delete} w_{insert} w_{rename}

Comparison

Jaccard

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

BLEU

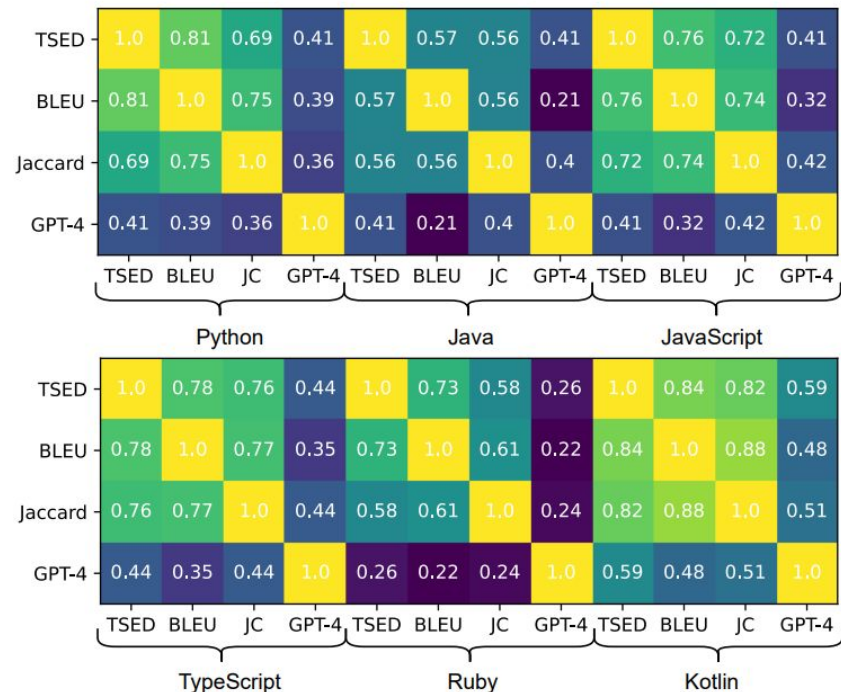
Text similarity (use for traduction scoring)

GPT-4

Ask if the codes are similar 0 to 1

TSED

$$\Delta(G_1, G_2) = \min_{ops} \sum_{i=1}^n w(op_i)$$



MBXP dataset

Pearson Correlation Heatmap between evaluation-metrics on GPT-3.5

Conclusion

TSED

Similar to BLEU

Fast (scalable)

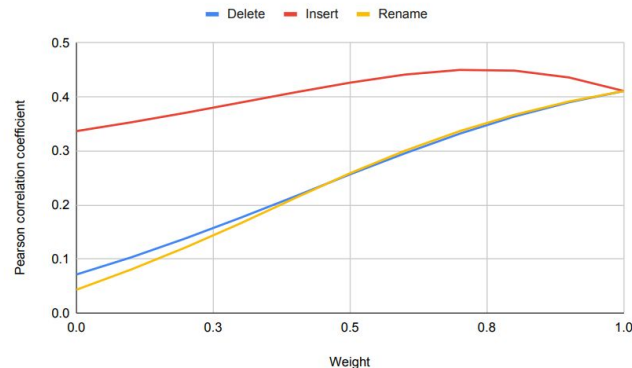
needs fine tuning

parsers dependent

	Python	Java	JavaScript	TypeScript	C#	Ruby	Kotlin
TSED	0.0227	0.0645	0.0315	0.0697	0.0373	0.0092	0.0307
BLEU	0.0075	0.0113	0.0155	0.0163	0.0160	0.0116	0.0144
Jaccard	1.6e-5	2.9e-5	1.9e-5	2.4e-5	2.7e-5	1.5e-5	1.8e-5
GPT3.5B Score	1304	1860	1231	1339	1470	1044	1681

Average execution time(ms) of metrics and programming languages

Metrics	1st	2nd	3rd	4th
Mean Squared Error	0.0581	0.0583	0.0527	0.0628
Mean Absolute Error	0.1902	0.1940	0.1825	0.1996



Lets talk

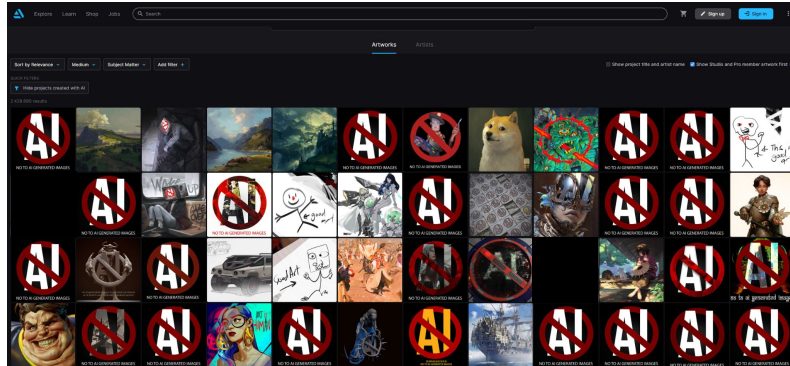
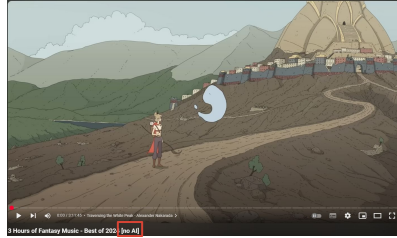
Problematic



Pinterest [Construct]

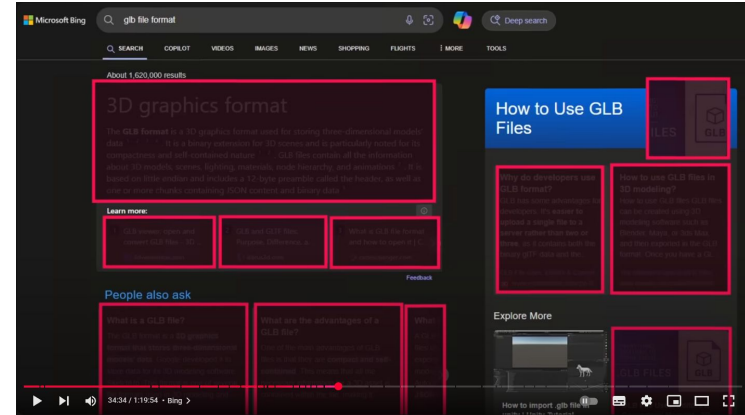


Problematic



Artists protest against AI generated content on Artstation

Freyja Holmér : Generative AI is a Parasitic Cancer



Media can be flooded by generative AI

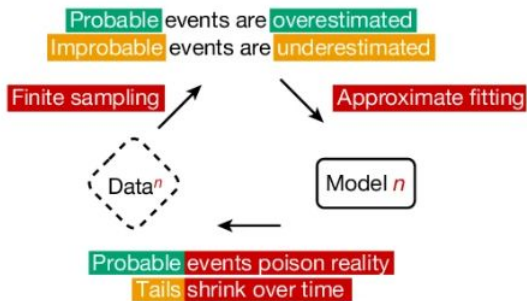
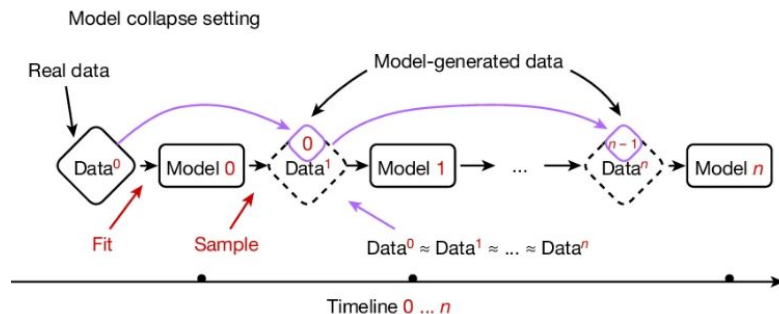
Generally cheap / poor content

Moderated by communities

ProblematicS

AI models collapse when trained on recursively generated data

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson & Yarin Gal



Model : OPT-125m (Meta)

Fine-tuning : wikitext2

Input : some started before 1360 — was typically accomplished by a master mason and a small team of itinerant masons, supplemented by local parish labourers, according to Poyntz Wright. But other authors reject this model, suggesting instead that leading architects designed the parish church towers based on early examples of Perpendicular.

Outputs:

Gen 0: Revival architecture such as St. John's Cathedral in London. The earliest surviving example of Perpendicular Revival architecture is found in the 18th @-@ century Church of Our Lady of Guernsey, which dates from the late 19th century. There are two types of perpendicular churches : those.

Gen 1: architecture such as St. Peter's Basilica in Rome or St. Peter's Basilica in Buenos Aires. There is no evidence that any of these buildings were built during the reign of Pope Innocent III, but it is possible that they may have been built during the reign of his successor, Pope Innocent.

Gen 5: ism, which had been translated into more than 100 languages including English, French, German, Italian, Spanish, Portuguese, Dutch, Swedish, Norwegian, Polish, Hungarian, Slovak, Lithuanian, Estonian, Finnish, Romanian, Bulgarian, Turkish, Croatian, Serbian, Ukrainian, Russian, Kazakh, Kyrgyz.

Gen 9 : architecture. In addition to being home to some of the world's largest populations of black @-@ tailed jackrabbits, white @-@ tailed jackrabbits, blue @-@ tailed jackrabbits, red @-@ tailed jackrabbits, yellow @-.