Plagiarism Detection Systems for Programming Assignments: Practical Considerations

Maxim Mozgovoy & Evgeny Pyshkin

The University of Aizu

{mozgovoy, pyshe}@u-aizu.ac.jp

Hello!

Primary area: *human-like AI for computer games and simulations*.

Other projects: *mostly educational technology-related*

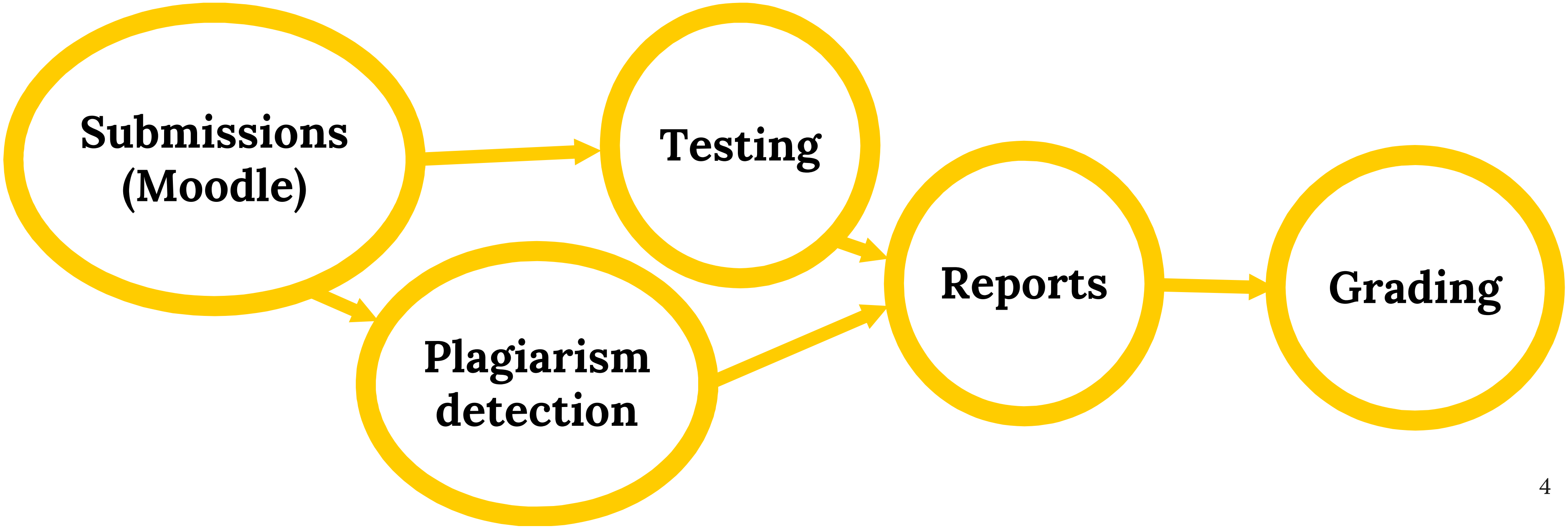
Note: this presentation is about a work-in-progress project!

Background

- Online teaching tools are in use for long time.
- This year much online teaching is more “streamlined” due to the pandemic.
- At our institution most people use the same combination of tools (Moodle / Zoom / etc.)
- Common scenarios can be revealed and automated.

Programming assignments assessment

One common pattern we see is this:



!

Main Observation

Automation of individual elements of this pipeline is well known, but it seems there is no comprehensive solution for a teacher to organize the whole process!

Main Idea

Basically, we want to create such a system. Our current main focus on the “plagiarism detection” line.

CSE: distinctive features

- “Master” code collections are typically available offline (unless we discuss classic algorithms).
- Students receive similar tasks and borrow from predecessors or peers.
- Code is easy to modify and obfuscate.

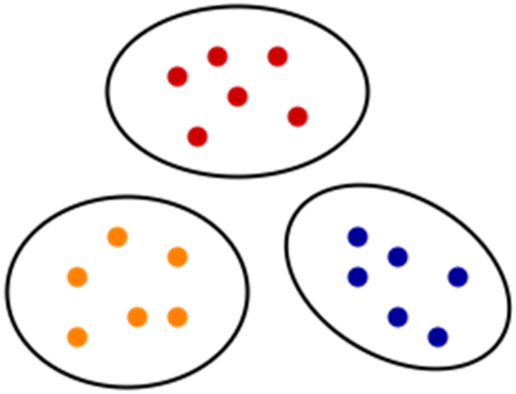
Facts & Challenges

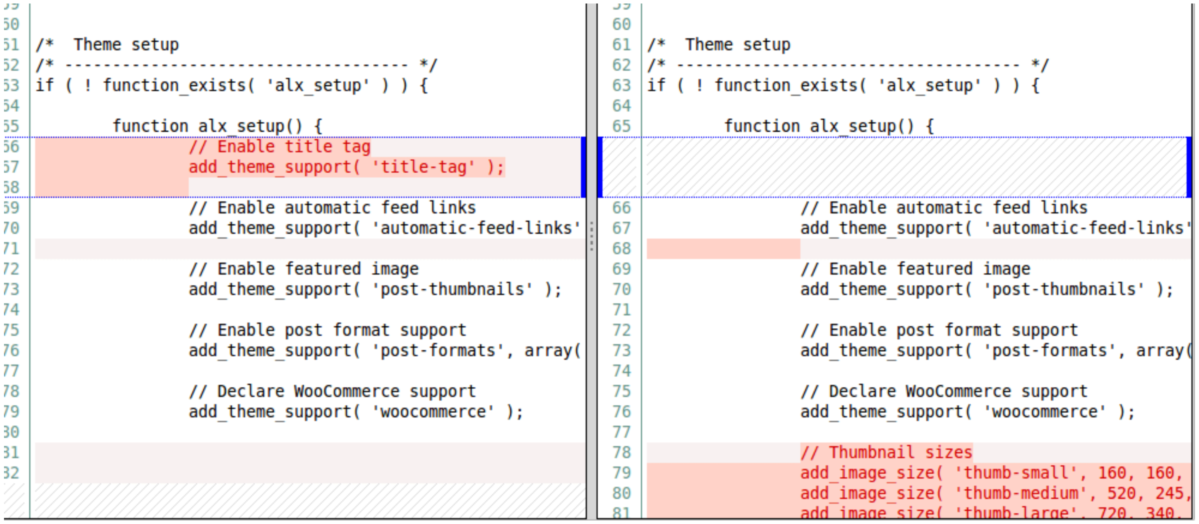
- Many relevant algorithms are already developed and available.
- However, most research works emphasize theory rather than daily teaching practice.
- Thus, we need a solution based on real everyday use cases.
- Since most time will be spent in analyzing reports, UI-related tasks are especially important and challenging.

Goals

- Retrieving data from Moodle (REST API is available).
- Comparing current submissions with peers' works and with historical submissions.
- Exclusion of code templates. Tokenization / languages support.
- Use of known similarity detection tools such as Plaggie or JPlag.
- Extensive reporting capabilities (e.g., clustering similar submissions)

Clustering & Similarity comparison





```
/*  Theme setup
/* ------------------------------------ */
if ( ! function_exists( 'alx_setup' ) ) {

        function alx_setup() {
                // Enable title tag
                add_theme_support( 'title-tag' );

                // Enable automatic feed links
                add_theme_support( 'automatic-feed-links'

                // Enable featured image
                add_theme_support( 'post-thumbnails' );

                // Enable post format support
                add_theme_support( 'post-formats', array(

                // Declare WooCommerce support
                add_theme_support( 'woocommerce' );


```

```
/*  Theme setup
/* ------------------------------------ */
if ( ! function_exists( 'alx_setup' ) ) {

        function alx_setup() {
                // Enable automatic feed links
                add_theme_support( 'automatic-feed-links'

                // Enable featured image
                add_theme_support( 'post-thumbnails' );

                // Enable post format support
                add_theme_support( 'post-formats', array(

                // Declare WooCommerce support
                add_theme_support( 'woocommerce' );

                // Thumbnail sizes
                add_image_size( 'thumb-small', 160, 160,
                add_image_size( 'thumb-medium', 520, 245,
                add_image_size( 'thumb-large', 720, 340,
```

Summary

- While there is not much novelty in our proposal, it would be of great help in practical situations.
- Available tools (except relevant GUI solutions) are basically available, but have to be brought together.
- Some issues like multi-language support or tokenization require further study, since quality of detection is still imperfect.

Thanks!

Contact me:
Maxim Mozgovoy
mozgovoy@u-aizu.ac.jp