Plagiarism Detection

Lucia D. Krisnawati

Overview

- Introduction to Automatic Plagiarism Detection (PD)
- External PD
- Intrinsic PD
- Evaluation Framework

Have You Heard, Seen or Used These?

turnitin®

PLAG AWARE

PlagScan
Our partner for academic integrity

URKUND

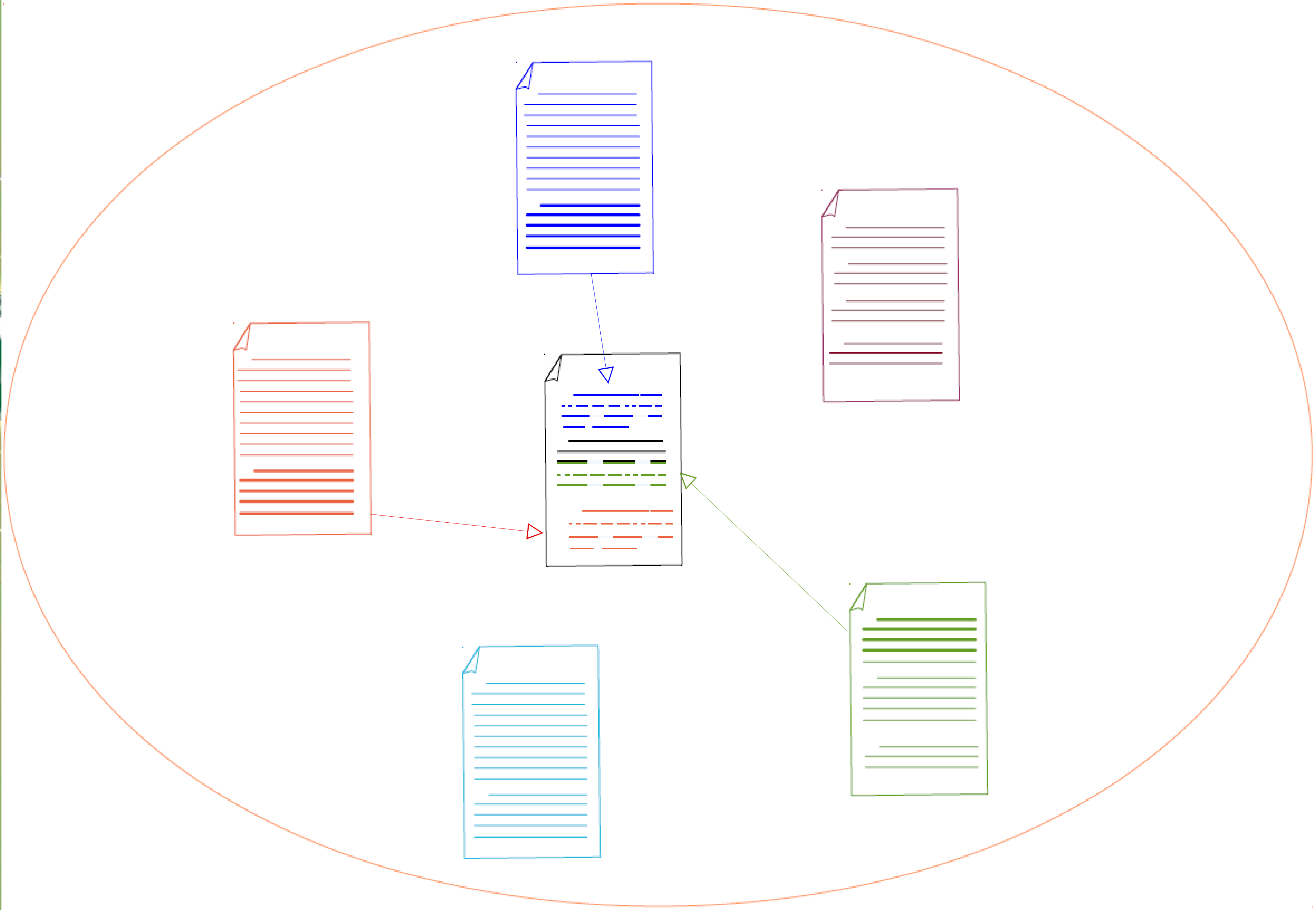
The Available PD Applications

- Turnitin:
  - The most widely distributed application of originality detection
  - Offline comparison to its vast database:
    - 45 billion pages,
    - 337 million millions in student archive
    - 130.000+ articles from academic & commercial journals & publications
  - Available in many European languages,
  - Asian languages: Chinese, Japanese, Korean
- other PD applications:
  - PlagAware, Plagiate Finder (German)
  - Viper
  - PlagScan
  - Etc.

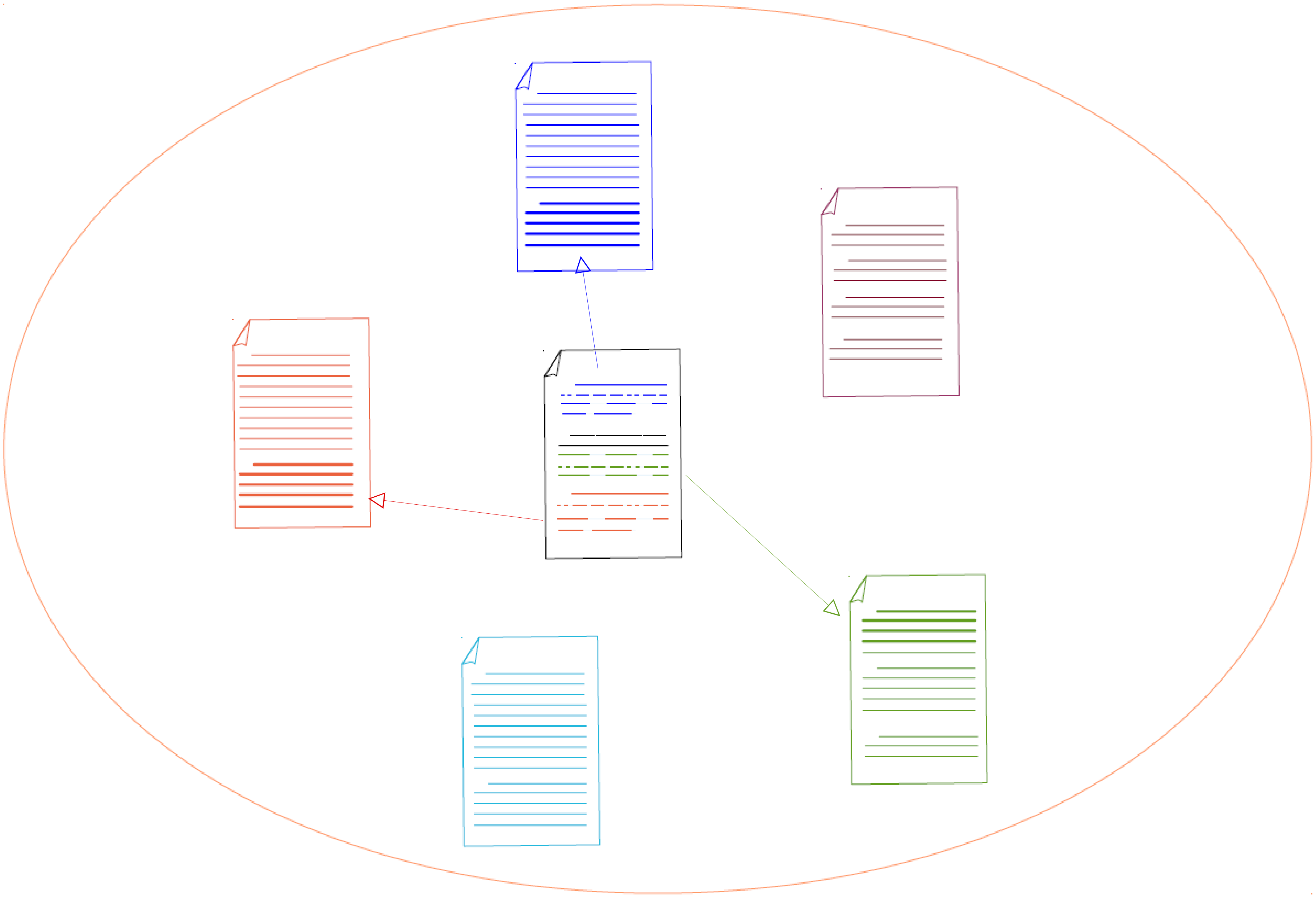
Why is Automated Plagiarism Detection?

- Plagiarism is a 'crime' in academic life
- The abundant availability of information in the Internet:
  - Easy access to research reports & findings (articles, etc)
  - Tremendous amount of source documents
    - Beyond human capacity to check
- IEEE defines plagiarism as:
  - „the reuse of someone else's prior ideas, results or words without acknowledging the original author & source“.

What is Plagiarism?



What is Plagiarism Detection?



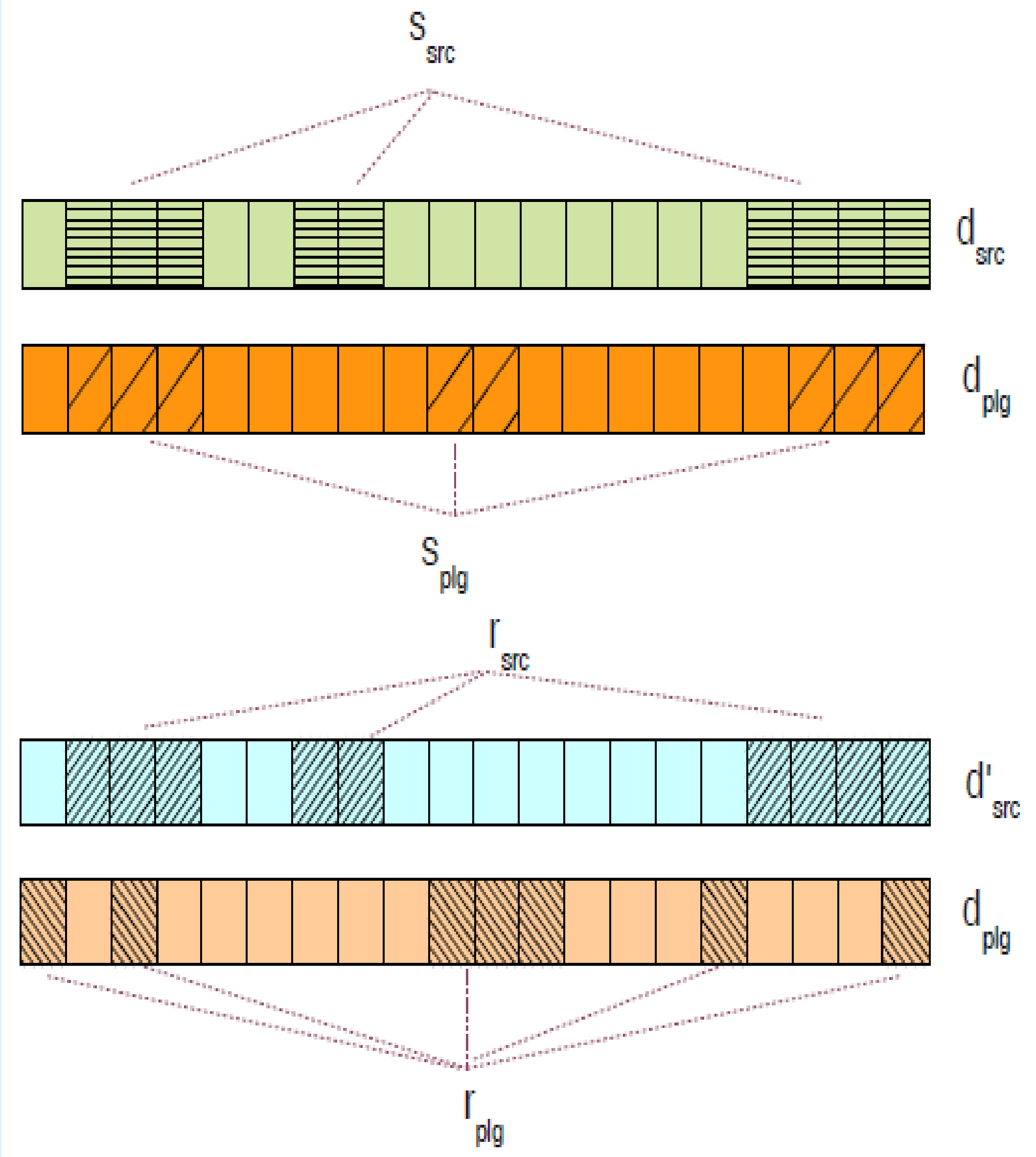
Automated Plagiarism Detection

- Formal Definition :
  - $S = \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle$ where s_{plg} is a passage in d_{plg} which is a plagiarized version of s_{src} in d_{src}
  - $r = \langle r_{plg}, d_{plg}, r_{src}, d'_{src} \rangle$ is a corresponding plagiarism detection
  - r is said to detect s iff:

    $s_{plg} \cap r_{plg} \neq \emptyset$,

    $s_{src} \cap r_{src} \neq \emptyset$, and

    $d_{src} = d'_{src}$
- Its task: finding out plagiarized portions within a piece of text



Quizzes

- Does near-duplicate detection share some commonolities with plagiarism detection? In what way?
- And what are their differences then?

Plagiarism Cases: Problems for APD

- Types of plagiarism cases:
  - Verbatim plagiarism
    - The direct copying of a passege from a source document
  - Disguised plagiarism
    - Shake & paste:
      - Done by removing, adding & replacing words/phrases
    - Paraphrasing or rewriting short parts of the passage
      - affecting its syntax
    - Technical disguise:
      - Refers to techniques that exploit weaknesses of curent detection methods to make plagiarized content non-machine detectable

        eg.
        - substituting characters with graphically identical symbols from foreign alphabets
        - Inserting random letters in white font

Plagiarism Cases: Problems for APD

- Types of plagiarism cases:
  - Cross-language plagiarism
    - The ideas are taken from the source document in different language.
    - A translated copy.
  - Idea plagiarism:
    - The use of broader concept without due to acknowledgement of the source
  - Self-plagiarism
    - The partial or complete re-use of one's own writings without being justified
- Ideally an APD algorithm should deal with these types of plagiarism cases.

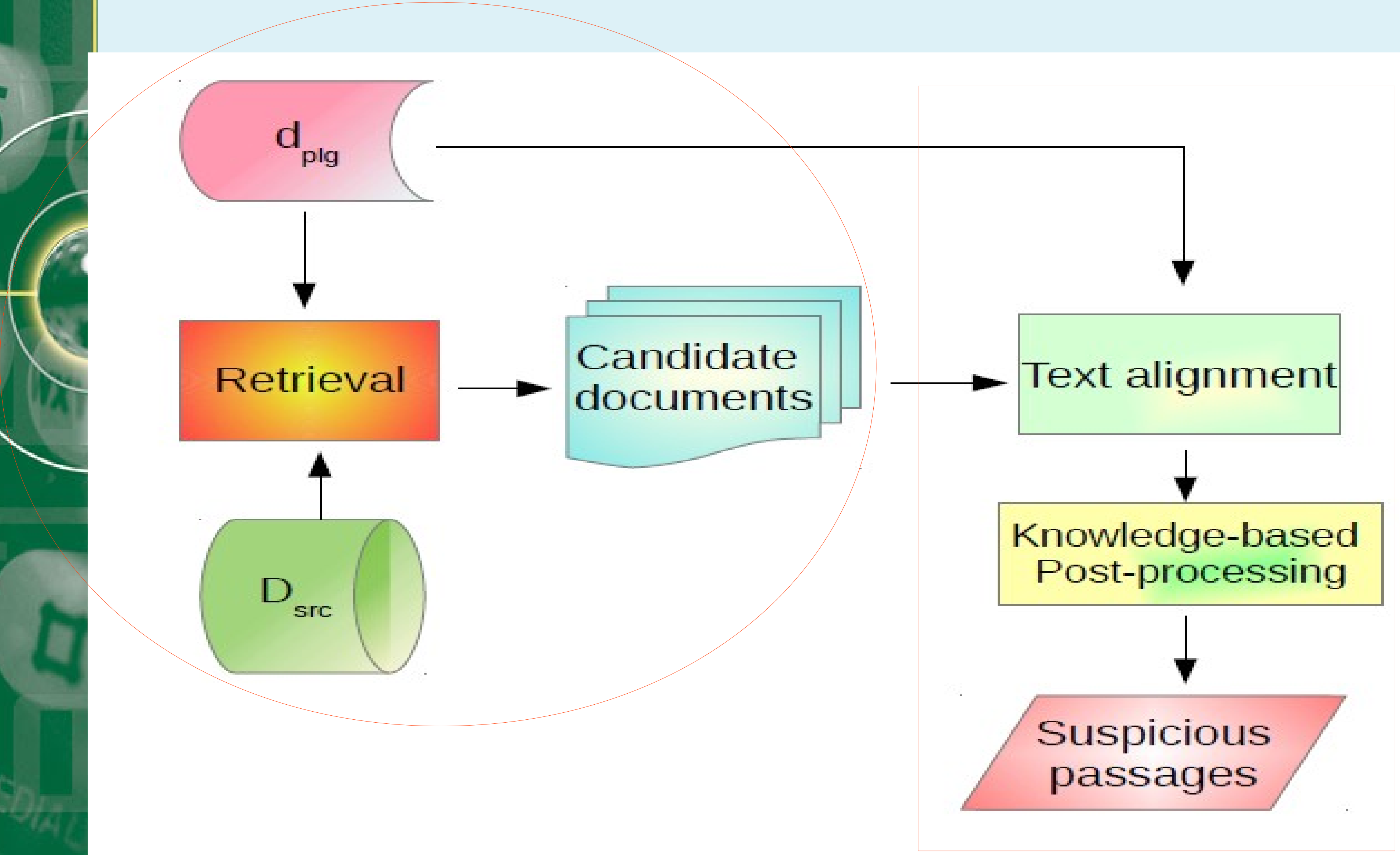
Automated Plagirasim Detection

- Based on its strategy, PD are classified into:
  - External Plagiarism
  - Intrinsic Plagiarism
- External Plagiarism Detection (EPD):
  - Given a d_{plg}, an algorithm in EPD attempts to detect *s* by retrieving d_{src} from a document collection D and by extracting s_{src} & s_{plg} from d_{src} & d_{plg} based on a detailed comparison between 2 documents.
- Intrinsic Plagiarism Detection (IPD):
  - IPD algorithm attemps to detect s by analyzing the writing style of $d_{plg.}$
  - Significant style variations indicate different writers

External Plagiarism Detection

A lot to do with IR

Beyond IR



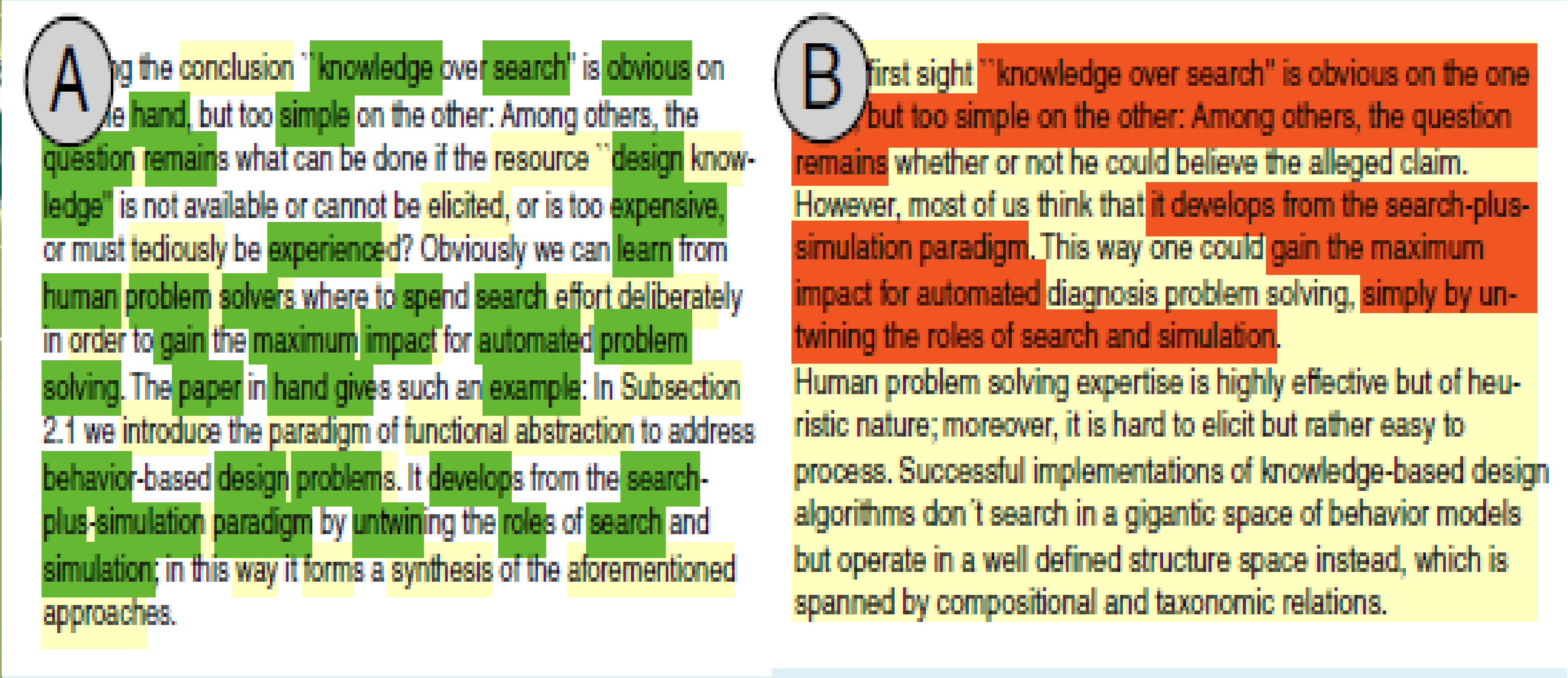
EPD Methods: Heuristic Retrieval

- Heuristic retrievel in EDP is a subtask which retrieves a relatively small subset of documents from the large corpus (web)
- The central challenges in heuristic retrieval:
  - How to keep the number of queries low
  - How to maximize recall

    This should be done simultaneously
- 2 common methods in heuristics-retrieval:
  - Following typical IR methods
  - Document fingerprinting

External Plagiarism Detection

- Which problems will we have if we just simply applying IR methods for Document similaritiy between suspicious and source documents?



A

ng the conclusion ``knowledge over search'' is obvious on
e hand, but too simple on the other: Among others, the
question remains what can be done if the resource ``design know-
ledge'' is not available or cannot be elicited, or is too expensive,
or must tediously be experienced? Obviously we can learn from
human problem solvers where to spend search effort deliberately
in order to gain the maximum impact for automated problem
solving. The paper in hand gives such an example: In Subsection
2.1 we introduce the paradigm of functional abstraction to address
behavior-based design problems. It develops from the search-
plus-simulation paradigm by untwining the roles of search and
simulation; in this way it forms a synthesis of the aforementioned
approaches.

B

first sight ``knowledge over search'' is obvious on the one
, but too simple on the other: Among others, the question
remains whether or not he could believe the alleged claim.
However, most of us think that it develops from the search-plus-
simulation paradigm. This way one could gain the maximum
impact for automated diagnosis problem solving, simply by un-
twining the roles of search and simulation.
Human problem solving expertise is highly effective but of heu-
ristic nature; moreover, it is hard to elicit but rather easy to
process. Successful implementations of knowledge-based design
algorithms don't search in a gigantic space of behavior models
but operate in a well defined structure space instead, which is
spanned by compositional and taxonomic relations.

EPD Methods: Heuristic Retrieval

- The common building blocks for retrieval algorithm with IR methods are:
  - Chunking
  - Keyphrase extraction
  - Query formulation
  - Search control
  - Download filtering

EPD Methods: Heuristic Retrieval

- **Chunking:**
  - A process of dividing a suspicious document into passages or chunks
  - Each chunk is processed individually
  - Goals :
    - to evenly distribute „attention“ over a suspicious document
    - To make the algorithm less susceptible to unexpected characteristics of the suspicious document
  - a non-overlapping chunk
  - Applied to suspicious document only
  - TextTiling:
    - A chunk method used to identify the topically related passage/ sentences

EPD Methods: Heuristic Retrieval

- **Chunking:**
  - Length of chunks:
    - The whole document as a chunk → no chunk
    - 50-line chunks
    - Paragraph chunking
    - 4-sentence chunks; 5-sentence chunks
    - 100-word chunks
  - Problems of line-based or paragraph-based chunks:
    - Failure detection for a non well-formatted documents
  - Larger chunk size:
    - Advantages:
      - more restrictive selectors → accuracy
      - Computationally more efficient → storing fewer chunks
    - Disadvantages:
      - Susceptible to failure in detetcting the disguised plagiairsm

EPD Methods: Heuristic Retrieval

- **Keyphrase extraction:**
  - Keyphrases are extracted to formulate queries
  - Rationale:
    - to select phrases which maximize the chance of retrieving source documents matching the suspicious document
    - To limit the amount of queries formulated
      - Reducing the cost of using the search engine
  - Output:
    - A single keyword per chunk
    - Phrase-query per chunk
    - 10- (15-, 20-) word query per chunk
  - This step is the most important one of a source retrieval:
    - Fewer keywords extracted
    - Better choice or recall is lost

EPD Methods: Heuristic Retrieval

- **Keyphrase extraction:**
  - Some methods:
    - Naive approach:
      - Simply extracting the first 10-grams per chunk
      - Using the longest sentence in the paragraph
    - Using syntactic analysis:
      - Extracting the first 3 disjunct sequential 10-grams
      - Only nouns, adjectives & verbs form the keyphrases
    - Using tf-idf scores:
      - Extracting the top-5, -10, -20 words scored by tf-idf per chunk
      - Combining the top-5 keywords with the most frequent 2 or 3 term collocations
    - Use the most unique 8-gram from each paragraph/ chunk
      - Uniqueness → words not contained in the previous chunks
    - PAT-tree & tf-idf
      - Extracting one 3-gram, two 4-gram & forty 5-gram with highest tf-idf scores that contain at least one of the top-10 tf-idf terms.

EPD Methods: Heuristic Retrieval

- **Query Formulation:**
  - The keywords from the chunk are formulated into queries
  - Applying filtering:
    - Formulating non-overlapping queries
    - Using the keyword once only per suspicious document
  - Tailoring keywords to the API of a search engine used .

EPD Methods: Heuristic Retrieval

- **Search Control:**
  - Given a set of queries, search controller:
    - Schedules their submission to the search engine
    - Directs the download of search result
  - Rationale:
    - To adjust dynamically the search based on the results of each query which may include:
      - Dropping queries
      - Reformulating existing ones
      - Formulaing new queries based on relevance feedback
  - Techniques:
    - Drop a query when more than 60% of its terms are contained in a previous downloaded document
    - Stop submitting queries when the number of queries exceeds the number of paragraphs in the suspicious documents

EPD Methods: Heuristic Retrieval

- **Download Filtering:**
  - Given a set of downloaded documents, the download filter removes all documents that are not worthwhile being compared in detail with the suspicious document
  - Rationale:
    - To further reduce the set of candidates
    - To save invocations of the subsequent detailed comparison
  - Some techniques used:
    - Focuses on the top-10 result of a query.
    - Downloading a result document when at least 90% of the words in a 160-character snippet contained in the suspicious documents
    - Consider the top-ranked documents and download it when at least 50% of query terms are contained in a 500 character snippet
    - Simply download the top-k result of the query
    - Download the top-3 documents whose snippets share at least five word 5-grams with the suspicious document

Heuristic Retrieval: Document Fingerprinting

- Global similarity measure:
  - analyzes characteristics of longer text sections or full text

    eg. Vector space model, stylometry
- Local Similarity measure:
  - Analyzes matches of confined text segments

    eg. document Fingerprinting
- Reasons for not or using Fingerprinting:
  - Fingerprinting is fast
  - But suffers on accuracy

Heuristic Retrieval: Document Fingerprinting

- An Example → Winnowing Algorithm:
  - Given a set of documents, we want to find the substring matches between them that satisfy 2 properties:
    - If there is a substring match at least as long as the guarantee threshold, t, then this match is detected
    - No detection of matches shorter than noise threshold k.
  - Note: k <= t and as k increases:
    - Matches less likely due to coincidence
    - Limit sensitivity to document component reordering
    - parameterization

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Step 1:
    - Eliminate context irrelevant features such as spaces, capitalization, and punctuation

Some text:

a). A do run run run, a do run run

The text with irrelevant features removed

b) Adorunrunrunadorunrun

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Step 2:
    - Choose appropriate t & k values
    - eg. t= 8 → detect any substring of at least 8
    - K = 5 → do't care about any substring of at least 8

Some text:

a). A do run run run, a do run run

the text with irrelevant features removed

b) Adorunrunrunadorunrun

the sequence of 5-gram derived from the text

c) adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Step 3:
    - Compute a rolling Rabin-Karp „style hash“ for every 5 character substring

Some text:

a). A do run run run, a do run run

the text with irrelevant features removed

b) Adorunrunrunadorunrun

the sequence of 5-gram derived from the text

c) adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun

a hypothetical sequence of hashes of the 5-gram

d) 77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Step 4:
    - Hashing string len 5 but care about strings len 8?
      - The pattern „adorunru“ is covered by (77, 74, 42, 17)
      - Hash & positional info from the start of string (77, 0) (74, 1)... (17, 3)
      - We only need to keep one from this „window“

Some text:

c) adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun

d) 77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98

e) (77, 74, 12, 17)   (74, 42, 17, 98)   (42, 17, 98, 50)
(17, 98, 50, 17)   (98, 50, 17, 98)   (50, 17, 98, 8)
(17, 98, 8, 88)   (98, 8, 88, 67) (8, 88, 67, 39)
(88, 67, 39, 77)   (67, 39, 77,74) (39, 77, 74, 42)
(77, 74, 42, 17)   (74, 42, 17, 98)

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Step 4 (cont.):
    - Given a set of hashes $h_1 ... h_n$ representing a document:
      - Let the window size w= t-k+1
      - Each possition 1 ≤ i ≤ n-w+1 in this sequence defines a window hashes $h_i ... h_{(i+w-1)}$
      - To maintain the guarantee of detection of all matches of length ≥ t, at least one of the h_i hashes must be chosen

Some text:

e) Window of hashes of length 4

(77, 74, 42, **17)**	(74, 42, 17, 98)	(42, 17, 98, 50)
(17, 98, 50, **17**)	(98, 50, 17, 98)	(50, 17, 98, **8**)
(17, 98, 8, 88)	(98, 8, 88, 67)	(8, 88, 67, 39)
(88, 67, **39**, 77)	(67, 39, 77,74)	(39, 77, 74, 42)
(77, 74, 42, **17**)	(74, 42, 17, 98)	

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm example:
  - Definition (Winnowing):
    - In each window select the minimum hash value. If there is more than one hash with the minimum value, select the rightmost occurance. Now save all selected hasheh as the fingerprints of the document.

  Some text:

  d) 77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98

  e) Window of hashes of length 4

(77, 74, 12, **17**)	(74, 42, 17, 98)	(42, 17, 98, 50)
(17, 98, 50, **17**)	(98, 50, 17, 98)	(50, 17, 98, **8**)
(17, 98, 8, 88)	(98, 8, 88, 67)	(8, 88, 67, 39)
(88, 67, **39**, 77)	(67, 39, 77,74)	(39, 77, 74, 42)
(77, 74, 42, **17**)	(74, 42, 17, 98)	

  f) fingerprints selected by winnowing

  17 17 8 39 17

  g) fingerprints paired with 0-base positional information

  [17, 3] [17, 6] [8, 8] [39, 11] [17, 15]

Heuristic Retrieval: Document Fingerprinting

- Winnowing algorithm
  - Why is this selection algorithm good?
    - Assume hashes are random number
    - Each window overlaps the previous window & only adds one new random number
    - The probabillity is low that when adding a new random number to a list that number is smaller than the ones already in that list
    - Thus, in practice need to store less than 1 value from each window

EPD: Text Alignment

- The goal of text alignment is:
  - To locate plagiarized content in the suspicious document along with its corresponding original text in source document
- The common building blocks for Text alignment are:
  - Seeding
  - Match merging/extension
  - Extraction filtering

EPD: Text Alignment

- Seeding Process:
  - Input: a suspicicous documents (d_{plg}) & output from retrieval process (d_{src})
  - Output: seeds matching d_{plg} & d_{src}
  - Seed heuristics identify either exact matches or create matches by changing:
    - the underlying texts in a domain-specific or
    - lingusitically motivated way.
  - The seeds can take form of:
    - Sentence pairs exceeded a given threshold
    - Sorted word N-grams, Sorted word-1-skip-N-grams
    - Unsorted stop word n-grams, Stop word n-gram
    - Name entity n-grams

EPD: Text Alignment

- Example of seeding process:
  - Sentence pair (Kong et. Al, 2012):
    - Segmenting both suspicious & source documents into sentences
    - Indexing sentences of source documents
    - Each sentence in suspicious document is used as a query to retrieve the index
    - Computing semantic similarity by cosine distance measure:

$$Sim(S,R) = \cos\theta = \frac{\sum_{k=1}^{n} w_{Sk} * w_{Rk}}{\sqrt{(\sum_{k=1}^{n} w_{Sk}^{2})(\sum_{k=1}^{n} w_{Rk}^{2})}} > t1$$

    - Where S= passage in d_{plg} , R is reference passage in d_{src} W_{Sk} & W_{RK} are weight of S & R; t_1 is threshold; $t_1 = 0.42$

EPD: Text Alignment

- Example of seeding process:
  - Sentence pair (Kong et. Al, 2012):
    - The next step is to compute the structure similarity of the sentence by:

$$T = \frac{2 * \sum_{t \in I_S \cap I_R} Min(N_{I_S}(t), N_{I_R}(t))}{|I_S| + |I_R|} > t2$$

    - Where Nis(t) & NIR(t) are number of overlapping terms, Min(NIs(t), NIR(t)) is the smallest one of Nis(t) & NIR(t), t_2 = 0.32
  - The sentence pairs that are in line with semantic & structural similarity are regarded as plagiarism candidate pairs

EPD: Text Alignment

- Example of seeding process:
  - Word 5-grams & stop word 8-grams (Suchomel, Kasprzak, & Brandej, 2013):
    - The features of document are formed by fingerprinting the word 5-gram and stop word 8-grams with MD5 algorithm
    - Two feature attributes that are saved: offset & length
    - Comparing the features of suspicious document to the source document
    - The matched features are saved for further process

EPD: Text Alignment

- Extension:
  - Input: seeds matches d_{plg} &d_{src}
  - Output: aligned text passages
  - Rationale for merging seeds are:
    - To determine whether a document contains a plagiarized passages rather than matching by chance
    - To identify a plagiarized passages as a whole rather than only its fragments
  - Most extension heuristics are rule-based and develop a set of constraints instead of 1 rule
  - Suchomel et al. employ a 2-step process of extension:
    - 1 step:
      - defining valid intervals consisting of at least 4 common features with maximum allowed gap inside set to 4000 characters
      - Or the adjacent seed matches that are less than 4000 chars apart are merged

EPD: Text Alignment

- Suchomel et al. employ a 2-step process of extension:
  - Step 2:
    - Criteria for joining the adjacent valid intervals:
      - The gap between interval contained at least 1 feature per 10,000 characters
      - The gap len < 30,000 chars & the size of adjacent interval is twice as the gap
      - The gap len < 30,000 chars & the number of common features per character in the adjacent interval is < 3 x the number of features per char in the possible joined interval.

EPD: Text Alignment

- Filtering:
  - Removing all aligned passages that do not meet certain criteria
  - Rationale are:
    - To deal with overlapping passages
    - To discard extremely short passages
  - Some rules for filtering:
    - Discarding word overlap whose jaccard coefficient value is below the threshold
    - Discarding passage length less than 190 or 300 characters
    - Discarding passages with less than 50 words
    - Discarding passages whose cosine similarity value below 0.75

Intrinsic Plagiarism Detection (IPD)

- The task of intrinsic plagiarism is:
  - To detect plagiarized passages of a suspicious document exclusively based on irregularities or inconsistencies within a document.
  - The inconsistencies are mainly stylistic nature
- The rationale:
  - Each writer has his own writing style
  - Changes between brilliant and baffling passages hint to a copy & paste or plagiarism
- The analysis is well known as stylometry.
- Stylometry is a field of research that attempts to quantify a writing style
- The quality of an intrinsic plagiarism detection depends on the quality of the quantified linguistic features

Intrinsic Plagiarism Detection

- The building blocks for IPD are:
  - Chunking strategy
  - Writing style retrieval model
  - An outlier detection algorithm
  - Postprocessing
- Chunking strategy:
  - Many researches employ a sliding window chunking with size ranging from 200-1000 words
  - The slide stepping of window ranges between 40-500 words
  - Each window will be compared to the rest of windows
  - Best performance: 400 & 1000 word chunk size

Intrinsic Plagiarism Detection

- Writing Style retrieval model:
  - is a model function that maps texts onto feature representations & their similarity measures.
  - Stamatatos(2009) distinguishes types of stylometric features into:
    - Lexical features → tf, word n-grams, vocabulary richness
    - Character features → character types, character n-grams
    - Syntactic features → POS frequency, types of phrases
    - Semantic features → synonym, semantic dependencies
    - Application-specific fetures → structural, content-specific, language specific

Intrinsic Plagiarism Detection

- Writing Style retrieval model:
  - Most researches in IPD use either lexical, character or syntactic features, eg.
    - A word vector including stop-words (Oberreuter, et al, 2011).
    - A binary vector including 100 rarest words that appear in at least 5% of all chunks (Akiva, 2011).
    - The 2500 most frequent character 3-grams (Kestemon, 2011)
  - Similarity measures used:
    - Cosine similarity
    - Stamatatos' normalized distance measure nd_1.

Intrinsic Plagiarism Detection

- Writing Style retrieval model:
  - Stamatatos' distance measure:

$$d_1(A,B) = \sum_{g \in P(A)} \left(\frac{2(f_A(g) - f_B(g))}{f_A(g) + f_B(g)} \right)^2$$

  - Where:
    - $f_A(g)$ & $f_B(g)$ are the frequency of n-gram g in text A & B;
    - P(A), P(B) are the profiles of text A & B.
  - d_1 is very stable even when the text length is diverse greatly.

Intrinsic Plagiarism Detection

- Writing Style retrieval model:
  - Stamatatos' normalized distance measure:

$$nd_1(A,B) = \frac{\sum_{g \in P(A)} \left(\frac{2(f_A(g) - f_B(g))}{f_A(g) + f_B(g)} \right)^2}{4|P(A)|}$$

  - Where:
    - $|P(A)|$ is the size of profile in text A.
  - The denominator ensures that the value of dissimilarity value lie between 0 and 1

Intrinsic Plagiarism Detection

- Outlier Detection:
  - Attemps to identify chunks that are noticeably different fromt the rest
  - 2 strategies applied:
    - Measuring the deviation from the average document style
    - Chunk clustering
  - Measuring the deviation from the average document style:
    - Rationale: to measure the chunk style that matches the average of d_{plg}.
    - Done by comparing each chunk representation with that of the whole chunks of d_{plg} .

Intrinsic Plagiarism Detection

- Outlier Detection:
  - Stamatatos' style change (sc) function for document D:

$$sc(i,D)=nd_1(w_i, D),\ i=1\ldots|w|$$

  - Where:
    - |w| is the total amount of windows.
    - l is the length of sliding window w
    - s is the slide stepping of window
    - x is the length of a text in characters
  - Given a text of x characters, |w| is computed:

$$|w| = \left\lfloor 1 + \frac{x - l}{s} \right\rfloor$$

Intrinsic Plagiarism Detection

- Chunk clustering:
  - Comparing the chunk representation
  - Attempting to cluster them into groups of similar styles
- Identifying plagiarized passages:
  - Given sc function, the task of an IPD is:
    - to detect peaks of that function corresponding to significantly different text section from the rest of the documents
  - Let: M → mean of sc (style change),

    S → sc standard deviation,

    a → a constant determining the sensitivity of plagiarism detection method, empirically determined to 2.0
  - The criterion for detecting plagiarized passage:

$$sc(i', D) > M' + a * S'$$

Intrinsic Plagiarism Detection

- Post-Processing:
  - Merging the overlapping and consecutive chunks that have been identified as outliers
  - Rationale: to decrease detection granularity

Evaluation Framework

- Evaluation Corpus
  - Most researches on PD build their own corpus
  - Only 1 standard corpus → Webis TRC-2013.
  - 2 kinds of plagiarism cases for evaluation corpus:
    - Simulated plagiarism case
    - Artificial plagiarism case
  - Corpus for source Documents in PlagDetect:
    - Web grabing from:
      - Kompas, Jurnal-ekonomi.org
      - http://www.karyatulisilmiah.com
      - http://artikel.staff.uns.ac.id
      - http://wartawarga.gunadarma.ac.id
      - http://carapedia.com

Evaluation Framework

- Corpus for Test Documents:
  - Creating simulated Plagiarism cases:
    - Done manually by purposeful modifications
    - Involving several writes
      - PlagDetect (my case): 34 writers
      - Webis-TRC-2013 : 27 writers
    - The procedure:
      - Plagdetect:
        - Writers can choose any topics that are poped-up on their pages
        - Modification types: paraphrasing, interleaving of 2 or more passages, deleting, inserting or simply copy & paste of passages from the source documents
        - Length of modification: at least 1 paragraph
      - Webis-TRC:
        - Given a topic, the writer uses a search engine (ChatNoir) to search for source material while preparing a document of 5700 words length on average.
        - Modification types: paraphrasing and interleaving of 2 or more passages

Evaluation Framework

- Corpus for Test Documents:
  - Creating artificial plagiarism (PlagDetect):
    - Done algorithmically
    - 2 obfuscation strategies:
      - Random text operation:
        - s_{plg} is created by shuffling, inserting, or deleting words at random
        - Insertion → using base-word lexicon
      - Semantic Word Variation:
        - s_{plg} is created by replacing words by one of their semantic categories
        - Using semantic_category lexicon

Evaluation Framework

- Performance measures:
  - S = a set of plagiarism cases in the corpus
  - R = a set of detections reported by a plagiarism detector
  - $s = \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle, s \in S$
  - Based on the notation above, precision & recall of R under S is measured as follows:

    Micro Precision & recall

$$prec_{\text{micro}}(S, R) = \frac{|\bigcup_{(s,r)\in(S\times R)}(\mathbf{s} \sqcap \mathbf{r})|}{|\bigcup_{r\in R}\mathbf{r}|},$$

$$rec_{\text{micro}}(S, R) = \frac{|\bigcup_{(s,r)\in(S\times R)}(\mathbf{s} \sqcap \mathbf{r})|}{|\bigcup_{s\in S}\mathbf{s}|},$$

$$\text{where} \quad \mathbf{s} \sqcap \mathbf{r} = \begin{cases} \mathbf{s} \cap \mathbf{r} & \text{if } r \text{ detects } s, \\ \emptyset & \text{otherwise.} \end{cases}$$

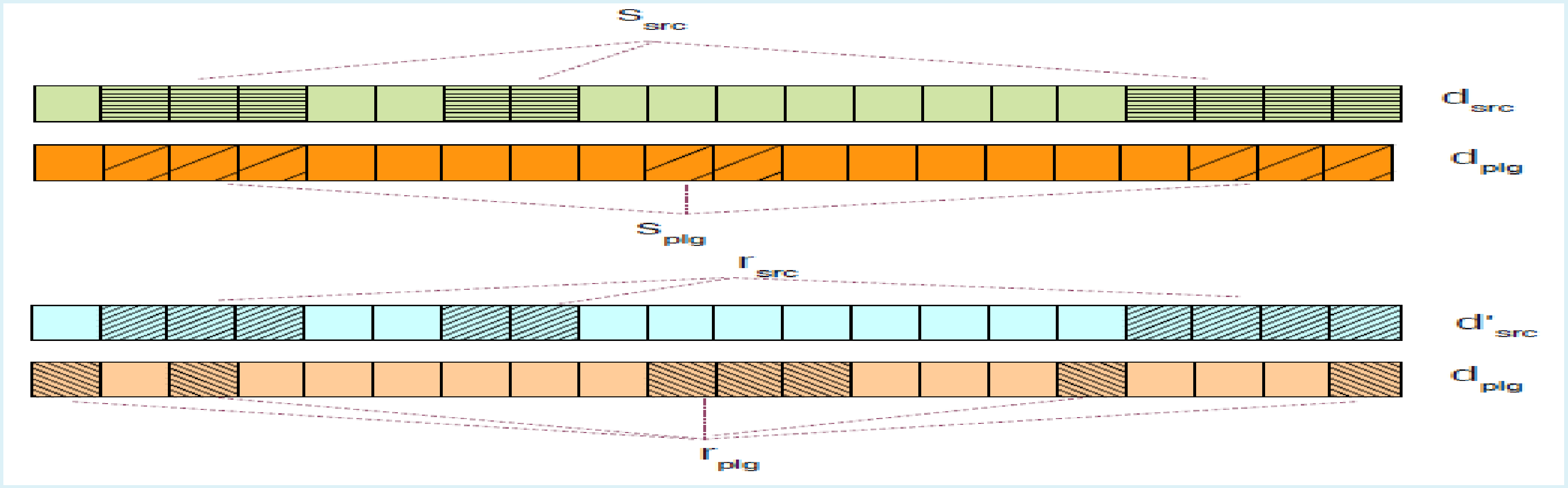
Evaluation Framework

- Performance measures:
  - Macro precision & recall:

$$prec_{\mathrm{macro}}(S,R) = \frac{1}{|R|}\sum_{r\in R}\frac{|\bigcup_{s\in S}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{r}|},$$

$$rec_{\mathrm{macro}}(S,R) = \frac{1}{|S|}\sum_{s\in S}\frac{|\bigcup_{r\in R}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{s}|},$$

Where s= s_{src} U s_{plg}; r = r_{src} U r_{plg}



References

- Meuschke, N., & Gipp, B. (2013). *State-of-the -Art in Detecting Academic Plagiarism.* In International Journal of Educational Integrity vol 9(1). pp 50-71.
- Potthast, M. et al. (2013). *Overview of the 5th International Competition on Plagiarism Detection*. In Forner, P et al (Eds). CLEF 2013 Evaluation Labs and Workshop.
- Potthast, M., Hagen, M., Völske, M & Stein, B. (2013). Crowdsourcing Interaction Log to Understand Text Reuse from the Web. In Proceedings of the 51st Annual Meeting of Assovciation for Computational Lingusitics, vol 1, pp 1212-1221.
- Schleimer, S., Wilkerson, D.S., & Aiken, A. (2003). *Winnowing: Local Algorithm for Document Fingerprinting*. In SIGMOD, ACM, pp. 76-85
- Stamatatos, E. (2011). *Plagiarism Detection Using Stopword n-grams.* Journal of the American Society for Information Science and Technology, 62(12). 2512-2527