Recent Trends in Digital Text Forensics and its Evaluation

Plagiarism Detection, Author Identification, and Author Profiling

Tim Gollub,[1] Martin Potthast,[1] Anna Beyer,[1] Matthias Busse,[1]
Francisco Rangel,[2,3] Paolo Rosso,[3] Efstathios Stamatatos,[4] and Benno Stein[1]

[1]Web Technology & Information Systems, Bauhaus-Universität Weimar, Germany
[2]Autoritas Consulting, S.A., Spain
[3]Natural Language Engineering Lab, ELiRF, Universitat Politècnica de València, Spain
[4]Dept. of Information & Communication Systems Engineering, University of the Aegean, Greece

`pan@webis.de` `http://pan.webis.de`

Abstract This paper outlines the concepts and achievements of our evaluation lab on digital text forensics, PAN 13, which called for original research and development on plagiarism detection, author identification, and author profiling. We present a standardized evaluation framework for each of the three tasks and discuss the evaluation results of the altogether 58 submitted contributions. For the first time, instead of accepting *the output* of software runs, we collected *the softwares themselves* and run them on a computer cluster at our site. As evaluation and experimentation platform we use TIRA, which is being developed at the Webis Group in Weimar. TIRA can handle large-scale software submissions by means of virtualization, sandboxed execution, tailored unit testing, and staged submission. In addition to the achieved evaluation results, a major achievement of our lab is that we now have the largest collection of state-of-the-art approaches with regard to the mentioned tasks for further analysis at our disposal.

1 Introduction

Nowadays, people increasingly share their work online, contribute to open projects and engage in web-based social interactions. The ease and the anonymity with which all of this can be done raises concerns about verifiability and trust: is a given text an original? Is an author the one who she claims to be? Does a piece of information originate from a trusted source? Answers to these and similar questions are crucial in order to deal with and rely on information obtained online, while the scale at which answers should be given calls for an automatic means. Specific tasks that address these questions include plagiarism detection, author identification, and author profiling, whereas tackling them requires expertise from diverse areas such as text forensics, computer linguistics, machine learning, and information retrieval, rendering research on these tasks a challenge.

Besides expertise, the research and development of solutions to these tasks is clearly limited due to the absence of representative evaluation resources and solid implementations of state-of-the-art approaches [18]. Moreover, researchers frequently lack the time and budget to acquire these resources themselves while researching their own approach. As a consequence, evaluations are often performed in an ad hoc manner, and only the data and approaches that are easily accessible are used to evaluate a new idea—a fact that impedes the comparability of published evaluation results significantly. This undesirable development can be mitigated by the development of standardized benchmarks and evaluation frameworks: in case of a widespread adoption by the community, individual researchers can compare their approaches independently, simply by following the evaluation guidelines of a given framework. However, the effort to develop and spread standardized evaluation frameworks is considerable, so that such frameworks emerge typically only for very popular tasks, whereas for the less studied tasks the quality depends on individual initiatives. To foster such initiatives, evaluation conferences are organized in order to bring together the stakeholders of a given task in the form of labs, where some develop a new evaluation framework, and others team up to develop approaches that are run against such a framework.

The typical modus operandi of such evaluation conferences can be summarized as follows: the organizers of a lab hand out a data set of instances of a task's underlying problem, which are downloaded and processed by the participants. They in turn compute and submit sets of solutions (so-called runs) to the problem instances, which are then evaluated by the organizers against an undisclosed gold standard of solutions. Beforehand, the organizers often hand out data sets comparable to the test data for training and development purposes. This process minimizes the "interface" between participants and organizers since they only need to agree on data formats. As a result, the evaluation resources provided by the task organizers may be used by other researchers later on, in order to compare their approaches against those of a lab's participants.

While organizing a lab this way requires least effort of all involved parties, there are also downsides with this approach: participants are not incentivized to *publish their software*, i.e., after the lab has passed other data sets cannot be evaluated by members of the community, and, the exact steps of how the participants obtained their results cannot be traced unless their approach is reimplemented. Taking into account that (1) even the best researchers make errors (including lab organizers), (2) devising an evaluation framework is a difficult engineering task, and (3) evaluation methodology evolves at rapid pace, the "classical" lab organization approach lacks long-term sustainability and reproducibility in first place.

In this paper we show how these shortcomings can be addressed in the context of digital text forensics: our contributions include the large-scale evaluation of 19 plagiarism detectors, 18 author identifiers, and 21 author profilers. Unlike traditional labs we do not collect software runs (outputs) but the softwares themselves, and evaluate them at our site. Our evaluation lab is the first that entirely switches to software submissions instead of run submissions. For this purpose we develop the TIRA experimentation platform, which facilitates such kinds of evaluations so that few staff can conduct the evaluation part-time. The outcome of our evaluation is not only a table of performance values and a data set, but also a collection of state-of-the-art implementations of a diversity of approaches to the three tasks. Given their original authors' consent, they can be readily used (via the web frontend of TIRA) by the community for comparison purposes, even on different data sets.

The remainder of this paper is organized as follows: after a brief discussion of related work, the Sections 2, 3, and 4 present insights into the evaluation results obtained for plagiarism detection, author identification, and author profiling respectively. Section 5 details our evaluation setup that handles the software submissions, and Section 6 draws conclusions.

1.1 Related Work

Before going into details, we review related work on evaluating plagiarism detectors, author identifiers, and author profilers, as well as related online evaluation platforms.

Plagiarism Detection In recent years, the evaluation of plagiarism and text reuse detectors has been studied in the context of the PAN evaluation labs that have been organized annually since 2009. For the purpose of these labs, we developed the first standardized evaluation framework which comprises a series of corpora of (semi-)automatically generated plagiarism as well as detection performance measures [43].[1] During the first three editions of the lab, a total of 43 plagiarism detectors have been evaluated using this framework [41, 42, 44]. The two recent editions refocused on specific sub-problems of plagiarism detection, namely source retrieval and text alignment. This also included the development of new corpora for these problems. Instead of again applying a semiautomatic approach to corpus construction, a large corpus of manually generated plagiarism has been crowdsourced in order to increase the level of realism [48]. This corpus comprises 297 essays of about 5000 words length, written by professional writers. In this regard the writers were given a set of topics to choose from along with two more technical rules: (1) to use the ChatNoir search engine [46] to research their topic of choice, and (2) to reuse text passages from retrieved web pages in order to compose their essay. The resulting essays represent the to-date largest corpus of realistic text reuse cases available, and they have been employed to evaluate another 33 plagiarism detectors in the past two labs [45, 47]. Besides the mentioned corpora, there are two other ones that comprise text reuse, namely the Meter corpus [9] and the Clough09 corpus [8]. The former contains 445 cases of text reuse among 1716 news articles, whereas the latter contains 57 short cases of manually generated plagiarism. To the best of our knowledge, these corpora have not yet been used in a large-scale evaluation of text reuse or plagiarism detectors.

Author Identification Author identification has many possible settings. Previous competitions on this task focused on closed-set and open-set classification problems with multiple candidate authors [2, 25, 26]. The evaluation corpora comprised a set of problems of similar form, i.e., a number of texts from a set of known authors and a number of texts of unknown authorship; the evaluation measures included traditional information retrieval measures such as micro- and macro-averaged accuracy, precision, recall and F_1. Author verification has been studied in the framework of the PAN 11 lab [2]. In contrast to our lab's setting, each problem comprised multiple test texts. Therefore, precision, recall, and F_1 per author (problem) were used for the evaluation of the participant methods. Accuracy and macro-average F_1 were also used in the evaluation of the well-known "unmasking" method [29]. In a recent work, Koppel and Winter [27] studied a similar problem where, given a pair of documents, the question

[1] The corpora PAN-PC-2009/2010/2011 are available at http://www.webis.de/research/corpora

is whether or not they are written by the same person. In addition to accuracy, they use recall-precision curves to provide a more complete picture of the performance of the examined models. Taking into account the nature of the practical applications involved with the task of author verification, it is crucial to estimate the ability of the attribution models to assign high confidence scores to their correct answers.

Author Profiling Our lab is the first to offer author profiling as an evaluation task. Therefore we review previous evaluations and data sets, where classification accuracy has been used in almost all cases as a performance measure. Pennebaker *et al.* [40] connected language use with personality traits, studying how the variation of linguistic characteristics in a text can provide information regarding gender and age of its author. Argamon *et al.* [3] analyzed formal written texts extracted from the British National Corpus, combining function words with part-of-speech features, and achieved approximately 80% accuracy in gender prediction. Other research investigated how to obtain age and gender information from formal texts [22, 7]. With the rise of the social media, Koppel *et al.* [28] built a dataset of blog posts and studied the problem of automatically determining an author's gender based on proposing combinations of simple lexical and syntactic features, also achieving approximately 80% accuracy. Schler *et al.* [51] collected more than 71,000 blog posts and used a set of stylistic features such as non-dictionary words, parts-of-speech, function words and hyperlinks, combined with content features, such as word unigrams with the highest information gain. They also obtained an accuracy of about 80% for gender identification, and about 75% for age identification. Goswami *et al.* [20] added some new features to Schler's work, such as slang words and the average length of sentences, improving accuracy to 80.3% in age group detection and to 89.2% in gender detection. Peersman *et al.* [38] compiled a dataset for the purpose of gender and age prediction from Netlog.[2] Studying short texts, Zhang and Zhang [58] experimented with segments of blog posts and obtained 72.1% accuracy for gender prediction. Similarly, Nguyen *et al.* [35] studied the use of language and age among Dutch Twitter users. They modelled age as a continuous variable (as they had previously done in [36]), and used a prediction approach based on logistic regression. They also measured the effect of gender in the performance of age detection, considering both variables as interdependent, and achieved correlations of up to 0.74 and mean absolute errors between 4.1 and 6.8 years.

Online Evaluation Platforms Based on our previous work in developing the TIRA experimentation framework [17, 18, 19], we revisit and update the related work. Our assessment of existing frameworks is based on the needs for local instantiation, web dissemination, platform independence, result retrieval, and peer to peer collaboration; Table 1 gives an overview. (1) The need for local instantiation arises from the fact that data may be kept confidential—i.e., the framework must be able to reside with the data instead of the other way around. External researchers then can use the service for comparison and evaluation of their own research hypotheses, whilst the experiment provider is in full control of the experiment resources. Apart from TIRA, this goal is currently only achieved by TunedIT. (2) Web dissemination is another important factor when developing an experimentation framework since it allows researchers to link the

[2] http://www.netlog.com

Table 1. Assessment of existing experimentation frameworks with respect to our five proposed design goals (1) local instantiation, (2) web dissemination, (3) platform independence, (4) result retrieval, and (5) peer to peer collaboration. The top six tools are non-commercial, developed out of universities, the bottom four are commercial ones.

Tool	**[Reference]**	**Domain**	**Design Goal**				
			1	2	3	4	5
evaluatIR	[5][1]	IR	×	✓	✓	✓	×
OpenML	[6][2]	ML	×	×	×	✓	×
MLComp	[3]	ML	×	✓	×	✓	×
myExperiment	[10][4]	any	×	✓	✓	✓	×
NEMA	[12][5]	IR	×	✓	×	✓	×
TunedIT	[57][6]	ML, DM	✓	✓	×	✓	×
TIRA	[19][7]	any	✓	✓	✓	✓	×
Google Code Jam	[8]	Algorithms	×	×	✓	✓	×
Kaggle	[9]	ML, DM	×	×	✓	×	×
TopCoder	[10]	any	×	×	✓	✓	×
Yahoo Pipes	[11]	Web	×	✓	×	×	×

[1]http://www.evaluatir.org
[2]http://www.openml.org
[3]http://www.mlcomp.org
[4]http://www.myexperiment.org
[5]http://www.music-ir.org
[6]http://www.tunedit.org
[7]http://tira.webis.de
[8]http://www.google.com/codejam
[9]http://www.kaggle.com
[10]http://www.topcoder.com
[11]http://pipes.yahoo.com

results in a paper with the experiment service used to produce them. Especially for standard preprocessing tasks or evaluations on private data, such a web service can become a frequently cited resource. However, not all frameworks currently pursue this goal. For example, Kaggle and TopCoder target commercial customers who typically refrain from sharing their assets, whereas Google Code Jam currently targets only scholars by organizing one-time competitions for education purposes. (3) The sophisticated and varying platform requirements of research experiments (as well as individual coding preferences of software developers) render the development constraints imposed by an experimentation framework critical for its success. Ideally, software developers can deploy experiments as a service that is unconstrained by the underlying operating system, parallelization paradigm, programming language, or data format. Local instantiation is a key to achieve this goal. Furthermore, the framework should operate as a layer on top of the experiment software and should use, instead of close intra-process communication such as in TunedIT, standard inter-process communication on the POSIX level to exchange information. (4) For computationally expensive retrieval tasks, the maintenance of a public result repository can become a valuable asset since it allows others to reuse them. Almost all frameworks support this goal. (5) Finally, by fostering peer-to-peer collaboration, a framework can drive a standardization process while maintaining a central repository of related evaluation resources. Note that currently none of the experimentation platforms implements peer-to-peer collaboration, though some have related functions.

2 Plagiarism Detection

This section briefly reports on the results of evaluating 18 plagiarism detectors that have been submitted to our evaluation lab. An extended version of this evaluation report can be found in [47], where a more in-depth analysis of the obtained results as well as a survey of detection approaches is given. To evaluate plagiarism and text reuse detectors, we measure their performance with regard to the two tasks source retrieval and text alignment, both of which are important parts of detectors that detect plagiarism from the web [54]. In the former task, a detector retrieves likely candidates from which text may have been reused in a suspicious document. In the latter task, the suspicious document is compared to selected candidates in closer detail. In the remainder of the section, we review the evaluation resources for each task individually and present the results of using it to evaluate the submitted detectors.

2.1 Source Retrieval

In source retrieval, given a suspicious document and a web search engine, the task is to retrieve all source documents from which text has been reused whilst minimizing retrieval costs. The cost-effectiveness of plagiarism detectors in this task is important since using existing search engines is perhaps the only feasible way for researchers as well as small and medium-sized businesses to implement plagiarism detection against the web, whereas search companies charge considerable fees for automatic usage. To study this task, we employ a controlled, static web environment, which consists of a large web crawl and search engines indexing it. Using this setup, we built a large corpus of manually generated text reuse in the form of essays, which serve as suspicious documents and which are fed into a plagiarism detector. The detection results returned are evaluated using tailored performance measures derived from precision and recall as well as cost-effectiveness statistics. Before discussing the actual performances obtained, we describe each of these resources in some detail.

Evaluation Setup Evaluating source retrieval in a reproducible, yet representative manner is a difficult endeavor, since this requires a search engine that indexes a representative portion of the web in a way so that the result sets of queries do not change, even after years. Commercial search engines are under constant development, so that they do not meet this constraint. Therefore, we resort to the current most representative research search engines Indri[3] and ChatNoir [46], which both index the ClueWeb09 corpus,[4] a 2009 web crawl of about one billion web pages, half of which are English ones. Since the ClueWeb corpus is static, the search engines that index it can be considered static as well, presuming their underlying retrieval models are not severely changed in the future. In order to independently measure the cost-effectiveness of source retrieval algorithms, we monitor access to the search engines by means of a central search proxy service. All source retrieval algorithms submitted to our lab used this service to retrieve sources for a given suspicious document. The service accepts search requests for Indri and ChatNoir and returns their search results in a unified format. Moreover, it serves

[3] http://lemurproject.org/clueweb09/index.php#Services
[4] http://lemurproject.org/clueweb09

web pages from the ClueWeb on demand. Besides unifying the search interfaces and result formats for the convenience of developers, all accesses to the search engines as well as the ClueWeb are logged minutely. This way, the performance of a source retrieval algorithm can be measured by analyzing the logs obtained after running it.

Evaluation Corpus As a realistic evaluation corpus we employ the Webis Text Reuse Corpus 2013 (Webis-TRC-13) [48]. The corpus has been constructed entirely manually and consists of 297 essays of about 5000 words length the contents of which have been reused from ClueWeb pages. The writers who wrote these essays were instructed to find web pages that match their respective essay's topic using the aforementioned ChatNoir search engine. If they decided to reuse a certain passage from a given web page, their instructions were to edit the reused text as thoroughly as they thought necessary to avoid detection. The modifications made include paraphrasing of the text itself as well as interleaving of reused passages from different sources. The average number of edits made on an essay is 2132.4, whereas the standard deviation is 1444.9. The average number of different sources used is 15.4, and the standard deviation 10. A subset of 40 essays of the Webis-TRC-13 was chosen as training documents, and 58 essays for testing. Based on this data, the source retrieval algorithms submitted to our lab were presented with a realistic retrieval setting, since it can be assumed that plagiarists as well as plagiarism detectors use the same search infrastructure to search for sources.

Performance Measures To assess the performance of a source retrieval algorithm, we measure its retrieval performance and the cost-effectiveness of obtaining its results. Retrieval performance is measured as precision, recall, and F_1 of retrieved sources regarding downloaded documents for a given suspicious document. The computation of precision and recall per suspicious document, however, is not straightforward, since each individual source of a given document may have a number of duplicates in the ClueWeb. These duplicates are not known a priori, so that each downloaded document has to be checked whether or not it is a duplicate of one of the sources of the suspicious document in question. If a downloaded document turns out to be a source duplicate, it is treated as a true positive detection (i.e., as if the original source had been found). However, retrieving more than one duplicate of a source document does not increase recall beyond that of retrieving just one, since no additional information is added by finding more duplicates of the same document. Conversely, retrieving more than one duplicate of a source document does not decrease precision, since they are not false positives. A detailed definition of what constitutes a source duplicate is beyond the scope of this overview, but can be found in [47].

Cost-effectiveness is measured as average workload per suspicious document, and as average numbers of queries and downloads until the first true positive detection has been made. These statistics reveal if a source retrieval algorithm finds sources quickly, thus reducing the costs of using it.

Evaluation Results Table 2 shows the performances of the nine plagiarism detectors that implemented source retrieval. Since there is currently no formula to organize retrieval performance and cost-effectiveness into an absolute order, the detectors are ordered alphabetically, whereas the best performance value for each metric is highlighted. As can be seen, there is no single detector that performs best on all accounts. Rather,

Table 2. Source retrieval results with respect to retrieval performance and cost-effectiveness.

Team (alphabetical order)	**Downloaded Sources**			**Total Workload**		**Time to 1st Detection**		**No Detection**	**Runtime**
	F_1	Precision	Recall	Queries	Downloads	Queries	Downloads		
Elizalde	0.17	0.12	0.44	44.50	107.22	16.85	15.28	5	241.7 m
Veselý	0.15	0.11	0.35	161.21	81.03	184.00	5.07	16	655.3 m
Gillam	0.04	0.02	0.10	16.10	33.02	18.80	21.70	38	**15.1 m**
Haggag	0.44	**0.63**	0.38	32.04	**5.93**	8.92	**1.47**	9	152.7 m
Kong	0.01	0.01	**0.65**	48.50	5691.47	2.46	285.66	**3**	4098.0 m
Lee	0.35	0.50	0.33	44.04	11.16	7.74	1.72	15	310.5 m
Nourian	0.10	0.15	0.10	**4.91**	13.54	**2.16**	5.61	27	25.3 m
Suchomel	0.06	0.04	0.23	12.38	261.95	2.44	74.79	10	1637.9 m
Williams	**0.47**	0.55	0.50	116.40	14.05	17.59	2.45	5	1163.0 m

different detectors have different characteristics. The detector of Williams *et al.* [56] achieves the best trade-off between precision and recall and therefore the best F_1 value. This detector is followed closely by that of Haggag and El-Beltagy [21], which achieves best precision but mediocre recall, whereas the detector of Kong *et al.* [31] achieves best recall at the cost of poor precision. It is not easy to decide which of these detectors solves the task best, since each of them may have their justification in practice. For example, the detector of Haggag and El-Beltagy downloads only about six documents on average per suspicious document and minimizes the time to first detection. Despite the excellent trade-off of Williams *et al.*'s detector, it incurs the second-highest costs in terms of queries on average, which is more than thrice as much as the other mentioned detectors. Kong *et al.*'s detector has highest download costs, but one may argue that downloads are much cheaper than queries, and that in this task recall is more important than precision.

2.2 Text Alignment

In text alignment, given a pair of documents, the task is to identify all contiguous passages of reused text between them. The challenge with this task is to identify passages of text that have been obfuscated, sometimes to the extent that, apart from stop words, little lexical similarity remains between an original passage and its plagiarized counterpart. Consequently, for evaluators, the challenge is to provide a representative corpus of documents that emulate this situation. To study this task, we employ a similar corpus construction methodology that has been used in previous evaluations of this task, while fixing some of its deficiencies. We evaluate the performance of plagiarism detectors based on the traditionally employed measures.

Evaluation Corpus The evaluation corpus for text alignment is also based on the aforementioned Webis-TRC-13. But instead of employing the essays of that corpus directly, pairs of documents that comprise reused passages have been constructed automatically, as was done in previous years [43]. One frequent point of criticism about automatically generating plagiarism is that it is difficult to ensure that documents between which text is plagiarized are about the same topic, so that the plagiarism could be detected simply by analyzing topic drift [48]. Using the documents that have been

Table 3. Text alignment results with retrieval performance and runtime.

Team	PlagDet	Recall	Precision	Granularity	Runtime
R. Torrejón	0.82220	0.76190	0.89484	1.00141	1.2 m
Kong	0.81896	0.81344	0.82859	1.00336	6.1 m
Suchomel	0.74482	0.76593	0.72514	1.00028	28.0 m
Saremi	0.69913	0.77123	0.86509	1.24450	446.0 m
Shrestha	0.69551	0.73814	0.87461	1.22084	684.5 m
Palkovskii	0.61523	0.53561	0.81699	1.07295	6.5 m
Nourian	0.57716	0.43381	0.94707	1.04343	40.1 m
baseline	0.42191	0.34223	0.92939	1.27473	30.5 m
Gillam	0.40059	0.25890	0.88487	1.00000	21.3 m
Jayapal	0.27081	0.38187	0.87901	2.90698	4.8 m

retrieved manually as sources for the essays of the Webis-TRC-13 as a basis for constructing plagiarism cases, however, allows us to mitigate this problem.

The corpus consists of pairs of documents about the same topic that share passages of text. These passages have been automatically obfuscated to emulate plagiarist behavior. We apply three basic obfuscation strategies, namely paraphrasing through naive random text operations and through cyclic translations, and summarization. Naive random text operations include shuffling, adding, removing, and replacing words at random while using WordNet as a source of word replacements and while optionally maintaining the original passage's part-of-speech sequence. Cyclic translations include, for example, translating a text from English to Japanese to Spanish and back to English using online translation services such as Google Translate. Summaries have been obtained by including an additional language resource from the Document Understanding Conference 2001 corpus for text summarization.[5] The corpus contains in total 1826 suspicious documents and 3169 source documents, which are grouped into 5000 pairs, so that there are 3000 pairs containing plagiarism (i.e., 1000 for each of the mentioned obfuscation strategies), 1000 containing unobfuscated plagiarism, and 1000 without plagiarism.

Evaluation Results Table 3 shows the overall performance of nine plagiarism detectors that implemented text alignment. The detailed performances of each detector with regard to different kinds of obfuscation can be found in [47]. Performances are measured using precision and recall at character level as well as granularity (i.e., how often the same plagiarism case is detected). These values are combined into the PlagDet score by dividing F_1 value of precision and recall by the granularity's logarithm. The two top-ranked detectors of Rodríguez Torrejón and Martín Ramos [49] and Kong *et al.* [31] achieve similar PlagDet scores, but differ in precision an recall. These detectors as well as that of Suchomel *et al.* [55] have been evaluated in previous years, all of which implement text alignment under the seed-and-extend paradigm: seeds, which encode positions of exact overlap between a pair of documents, are identified and then aligned into passages based on their pairwise distance. Examples for seeds include 5-grams, and stop word 8-grams [53].

[5] http://www-nlpir.nist.gov/projects/duc/data/2001_data.html

3 Author Identification

Authorship attribution is an important problem in many areas including information retrieval and computational linguistics, but also in applied areas such as law and journalism where knowing the author of a document (such as a ransom note) may be crucial to save lives. The most common framework for testing algorithms that solve this task is a closed-set text classification problem: given a sample of documents from a small, finite set of known candidate authors, the task is to determine for a document of unknown authorship, which author, if any, wrote the document in question [24, 52]. It has been commented, however, that this may be an unreasonably easy task [30]. A more demanding problem is author verification where, given a set of documents by a single author and a document of unknown authorship, the task is to determine if the document was written by that particular author or not [29]. This setting more accurately reflects real life in the experiences of professional forensic linguists, who are often called upon to answer this kind of question. Interestingly, every author identification problem with multiple candidate authors can be transformed to a set of author verification problems. Therefore, the ability to effectively deal with author verification is fundamental in author identification research.

Evaluation Setup The author identification task of our lab is set up as follows: given a small set (no more than 10, possibly as few as one) of "known" documents by a single person and an "unknown" document, the task is to determine whether the unknown document was written by the same person who wrote the known document set. The participants were given several problems of this form in three natural languages: English, Greek, and Spanish. One problem comprises a set of known documents by a single person and exactly one unknown document. The number of known documents per problem varies from 1 to 10. All documents within a single problem are in the same language, and best efforts were applied to assure that within-problem documents are matched for genre, register, theme, and date of writing. Moreover, the length of the documents varies from a few hundred to a few thousand words. The participants were asked to develop their software so that they can handle any set of such author verification problems in the specified languages. For each problem, they have to generate a binary answer ("yes", if the unknown document was written by that author or "no", if the unknown document was not written by that author). It was also possible to leave some problems unanswered. In addition, the participants could optionally produce a confidence score, namely a real number in the interval $[0, 1]$ where 1 means that it is absolutely sure that the unknown document was written by that author and 0 means the opposite.

Evaluation Corpus The corpus we built for the author identification task covers three languages: English, Greek, and Spanish. For each language there is a set of problems, where one problem comprises a set of documents of known authorship by a single author and exactly one document of unknown authorship. All the documents within a problem are in the same language, placed in a separate folder, and the language information was encoded in the problem label (i.e., folder name). The training corpus comprised 10 problems in English, 20 problems in Greek and 5 problems in Spanish. The test corpus was more balanced across languages comprising 30 problems in

English, 30 problems in Greek and 25 problems in Spanish. The English part of the corpus[6] consists of extracts from published textbooks on computer science and related disciplines. The Greek part of the corpus comprises newspaper articles published in the Greek weekly newspaper TO BHMA[7] from 1996 to 2012. The Spanish part of the corpus[8] consisted of excerpts from newspaper editorials and short fiction.

Performance Measures The participants of our lab were asked to provide a simple "yes/no" binary answer for each problem of the author identification task. Optionally, in case a software was not confident enough for to decide a problem, it could be left unanswered. To evaluate the output of a software, we used the following measures:

$$\text{Recall} = \frac{\#\text{correct\_answers}}{\#\text{problems}} \qquad \text{Precision} = \frac{\#\text{correct\_answers}}{\#\text{answers}}$$

Note that in case a participant's software provides answers all problems, these two measures are equal.

The final ranking was computed by combining these measures via F_1 for the whole evaluation corpus comprising all three languages. That way, a method that can only deal with a certain language will be ranked very low. In addition, to evaluate the participants that also submitted a confidence score (a real number in the set $[0, 1]$) we used Receiver-Operating Characteristic (ROC) curves and the area under the curve (AUC) as a single measure. ROC curves provide a more detailed picture over the ability of the author verification methods to assign high confidence scores to their answers. For the calculation of ROC curves, any missing answers were assumed to be wrong answers. Again, softwares that can only handle documents of a certain language will produce low AUC scores. Finally, since we asked for software submissions so that the software is executed at our site, it is possible for the first time to compare the runtime of the different author verification methods.

Evaluation Results In total, 18 participants submitted their software for this task. The final evaluation results and the ranking of the participants according to the overall F_1 score are depicted in Table 4 (left). Results for each of the three examined languages are provided as well. Moreover, 10 participants also submitted confidence scores together with their binary answers. This allowed us to compute ROC curves and the corresponding AUC values for those participants. The results of this evaluation are shown in Table 4 (right).

As concerns the features to represent the stylistic properties of texts, traditional solutions were followed including mainly character, lexical, and syntactic features. The latter require the use of language-specific NLP tools and considerably increase the runtime cost. The classification methods can be divided into intrinsic and extrinsic ones. Intrinsic methods make their decisions based solely on the set of known and unknown documents per problem. Conversely, extrinsic methods use external resources, such as additional documents of known authorship taken from the training corpus or downloaded

[6] This part of the corpus was contributed by Patrick Brennan of Juola & Associates.

[7] http://www.tovima.gr

[8] Sheila Queralt of Universitat Pompeu Fabra and by Angela Melendez of Duquesne University assisted in preparing this part of the corpus.

Table 4. Author identification results in terms of F_1 and runtime (left table) as well as AUC for softwares that output confidence scores (right table).

Team	Overall	English	Greek	Spanish	Runtime
Seidman	0.753	0.800	0.833	0.600	1091.3 m
Halvani	0.718	0.700	0.633	0.840	0.1 m
Layton	0.671	0.767	0.500	0.760	0.2 m
Petmanson	0.671	0.667	0.567	0.800	603.6 m
Jankowska	0.659	0.733	0.600	0.640	4.0 m
Vilarino	0.659	0.733	0.667	0.560	93.0 m
Bobicev	0.655	0.644	0.712	0.600	28.6 m
Feng	0.647	0.700	0.567	0.680	1406.9 m
Ledesma	0.612	0.467	0.667	0.720	0.5 m
Ghaeini	0.606	0.691	0.461	0.667	2.1 m
van Dam	0.600	0.600	0.467	0.760	0.2 m
Moreau	0.600	0.767	0.433	0.600	130.0 m
Jayapal	0.576	0.600	0.633	0.480	0.1 m
Grozea	0.553	0.400	0.600	0.680	6.8 m
Vartapetiance	0.541	0.500	0.533	0.600	7.0 m
Kern	0.529	0.533	0.500	0.560	10.4 m
baseline	0.500	0.500	0.500	0.500	–
Veenman	0.417	0.800	–	–	16.0 m
Sorin	0.331	0.633	–	–	60.7 m

Team	Overall	English	Greek	Spanish
Jankowska	0.777	0.842	0.711	0.804
Seidman	0.735	0.792	0.824	0.583
Ghaeini	0.729	0.837	0.527	0.926
Feng	0.697	0.750	0.580	0.772
Petmanson	0.651	0.672	0.513	0.788
Bobicev	0.642	0.585	0.667	0.654
Grozea	0.552	0.342	0.642	0.689
baseline	0.500	0.500	0.500	0.500
Kern	0.426	0.384	0.502	0.372
Layton	0.388	0.277	0.456	0.429
Sorin	0.082	0.658	–	–

from the web, and usually attempt to transform the one-class classification problem to a binary classification problem. The winning submission follows this approach and is based on the impostors method introduced in [27]. Ensemble classification models are very effective in both intrinsic and extrinsic approaches. Most participants attempt to tune the parameters of their systems separately for each language and sometimes they use external corpora in this procedure. Moreover, text length normalization seems to be a significant factor especially for producing a reliable confidence score for each provided answer.

4 Author Profiling

Author profiling is about predicting an author's demographics based on her writing. For example, profiling algorithms are used to determine an author's gender, age, native language, personality type, etc. Author profiling is a problem of growing importance in a variety of areas, such as forensic linguistics and marketing. From the former perspective, the ability to determine the linguistic profile of the author of a suspicious text solely by analyzing the text is useful for suspect verification. Similarly, from a marketing perspective, companies are interested to know what types of people like or dislike their products, based on the analysis of blogs and online product reviews.

The starting point for our research is the seminal work of Argamon et al. [4], who were the first to demonstrate a correlation of word usage and author demographics. Until now, however, research within computational linguistics [3] and social psychology [39] has mainly focused on English text. In our lab, we therefore focus on predicting an author's gender and age based on both English and Spanish text. Moreover, we put

Table 5. Corpus statistics of the evaluation corpus applied for author profiling.

Lang	Age	Gender	No. of Authors	
			Training	Test
en	10s	male	8 600	888
		female	8 600	888
	20s	male	(72) 42 828	(32) 4 576
		female	(25) 42 875	(10) 4 598
	30s	male	(92) 66 708	(40) 7 184
		female	66 800	7 224
Σ			236,600	25,440

Lang	Age	Gender	No. of Authors	
			Training	Test
es	10s	male	1 250	144
		female	1 250	144
	20s	male	21 300	2 304
		female	21 300	2 304
	30s	male	15 400	1 632
		female	15 400	1 632
Σ			75 900	8 160

particular emphasis on the use of everyday language and analyze how it reflects basic social and personality processes by using text obtained from social media.

Evaluation Corpus To construct a large-scale evaluation corpus, we crawled public social media sites where user posts can be obtained along with labels that indicate author demographics such as gender and age. Table 5 shows the basic statistics of the compiled English and Spanish corpora. The corpora consist of files where each file contains at least one post and at most 1000 words of combined posts of an individual author. In case an author wrote posts amounting to more than that, more than one file for that author were generated. Authors with little data were kept in order to provide a realistic cross-section of authors within our evaluation framework.

The age labels are divided into age groups, following the approach of Schler et al. [51]: 10s (ages 13-17), 20s (ages 23-27) and 30s (ages 33-47). Within each age group, the subcorpora are balanced by gender; however, the subcorpora between age groups were left unbalanced. In addition, we introduced a small number of posts from sexual predators as well as posts with a sexual topic obtained from conversations between adults [23]. The numbers in parentheses found in the table denote the number of such conversations in the respective parts of our corpus.

Evaluation Results In Table 6, the prediction accuracies for gender, age groups, and the combination are shown. Accuracies compute as ratio of the number of correctly predicted authors and total number of authors. To obtain the total score, we compute the average of the accuracies. The overall best performing approach across both languages was provided by Lopez-Monroy et al. [33], computed as averaged between both English and Spanish.

With regard to the used features among the different approaches Lopez-Monroy et al. [33] employ second-order representations based on relationships between documents and author profiles, whereas Meina et al. [34] exploit collocations. The latter do not seem to perform as good in Spanish as they do in English, or they are more difficult to be tuned. Almost all approaches rely on writing style features. Nevertheless, a wide variety of performances were obtained, showing that they may not be very easy to handle. Part-of-speech features were employed by five different approaches, including the two best performing ones for English [34] and Spanish [50], whereas the remaining systems are ranked below the median rank. Readability features are also widely used: the approach of Gillam [16] uses them exclusively, which demonstrates the performance

Table 6. Author profiling results in terms of accuracy on English (left) and Spansih (right) texts.

English			
Team	Total	Gender	Age
Meina	0.3894	0.5921	0.6491
Pastor L.	0.3813	0.5690	0.6572
Seifeddine	0.3677	0.5816	0.5897
Santosh	0.3508	0.5652	0.6408
Yong Lim	0.3488	0.5671	0.6098
Ladra	0.3420	0.5608	0.6118
Aleman	0.3292	0.5522	0.5923
Gillam	0.3268	0.5410	0.6031
Kern	0.3115	0.5267	0.5690
Cruz	0.3114	0.5456	0.5966
Pavan	0.2843	0.5000	0.6055
Caurcel Diaz	0.2840	0.5000	0.5679
H. Farias	0.2816	0.5671	0.5061
Jankowska	0.2814	0.5381	0.4738
Flekova	0.2785	0.5343	0.5287
Weren	0.2564	0.5044	0.5099
Sapkota	0.2471	0.4781	0.5415
De-Arteaga	0.2450	0.4998	0.4885
Moreau	0.2395	0.4941	0.4824
baseline	0.1650	0.5000	0.3333
Gopal Patra	0.1574	0.5683	0.2895
Cagnina	0.0741	0.5040	0.1234

Spanish			
Team	Total	Gender	Age
Santosh	0.4208	0.6473	0.6430
Pastor L.	0.4158	0.6299	0.6558
Cruz	0.3897	0.6165	0.6219
Flekova	0.3683	0.6103	0.5966
Ladra	0.3523	0.6138	0.5727
De-Arteaga	0.3145	0.5627	0.5429
Kern	0.3134	0.5706	0.5375
Yong Lim	0.3120	0.5468	0.5705
Sapkota	0.2934	0.5116	0.5651
Pavan	0.2824	0.5000	0.5643
Jankowska	0.2592	0.5846	0.4276
Meina	0.2549	0.5287	0.4930
Gillam	0.2543	0.4784	0.5377
Moreau	0.2539	0.4967	0.5049
Weren	0.2463	0.5362	0.4615
Cagnina	0.2339	0.5516	0.4148
Caurcel Diaz	0.2000	0.5000	0.4000
H. Farias	0.1757	0.4982	0.3554
baseline	0.1650	0.5000	0.3333
Aleman	0.1638	0.5526	0.2915
Seifeddine	0.0287	0.5455	0.0512
Gopal Patra	–	–	–

of such features in isolation. With the exception of Meina et al. [34]'s approach, all developers that employ n-gram features are also ranked below the median rank. Using sentiment words [37] and emotion words [13, 14] does not seem to improve accuracy in the same way as using slang words [1, 11, 13, 14]; however, these difference may be due to other features used by the same approaches. Finally we note that, with the exception of Lim et al. [32] and Meina et al. [34], all approaches that employ some kind of preprocessing on the corpora perform worse.

5 Handling Large-Scale Software Submissions with TIRA

This is the second time our lab accepts software submissions; in total, 58 softwares were submitted for the three tasks combined. This is more than five times as much compared to last year, where eleven softwares were submitted for the aforementioned plagiarism text alignment task [17]. Building on these experiences, we continue the development of the TIRA experimentation platform which serves as a valuable toolbox for organizing and managing our evaluation process.[9] In what follows, we outline the challenges of software submissions, discuss the technological and organizational means to meet them and how they are currently implemented within TIRA. Moreover, we present an analysis of user errors that provides insights into open problems and gives directions for future development.

[9] http://tira.webis.de

5.1 Challenges of Software Submissions, and our Solutions

In traditional evaluation labs, the lab organizers prepare and release a data set for a given task, withholding the ground truth data. Participants research and develop algorithms that solve the task and process the data at their site. Their algorithms' output (so-called runs) is submitted to the lab organizers who evaluate them against the ground truth. The only difference of our lab to the traditional process is that, instead of runs, we asked participants to submit their software in order for it to be run at our site and to be preserved in executable state for future evaluations. Accepting software submissions introduces a number of technical and organizational challenges, though. For each of these challenges, we devise tailored solutions:

1. *Environment Diversity.* With run submissions, participants are not limited with regard to their work environments (i.e., operating systems and programming languages). With software submission, lab organizers either need to restrict work environments or be prepared to execute arbitrary software.

   Our solution: virtualization; each participant gets full access to a virtual machine and deploys her software so that it can be executed via a pre-defined command.

2. *Executing Untrusted Software.* With software submissions, lab organizers are required to execute participant software at their site. The software often comes in the form of binaries instead of source code; in any case it is virtually impossible to ensure the trustworthiness of submitted software.

   Our solution: virtualization; virtual machines encapsulate submitted software.

3. *Data Leakage.* With software submissions, lab organizers may feed private data into the software. However, since the software is untrusted, this data may leak to the public via a number of channels that need to be monitored and secured by lab organizers.

   Our solution: sandboxing; before executing software, virtual machines are disconnected from the network, copied, and restored to their previous state afterwards.

4. *Error Handling.* With run submissions, participants debug their software directly. The only errors that may go unnoticed until after submission are errors in the run format specified by lab organizers. With software submissions, however, lab organizers may experience software errors because of insufficiently tested software or because of phenomena present in the test data that are absent from the training data.

   Our solution: unit testing; in case of errors, participants are notified by mail.

5. *Responsibility.* With software submissions, lab organizers assume partial responsibility for the successful evaluation of a participant's software. They must be vigilant about all kinds of errors that may invalidate the output of a submitted software.

   Our solution: staged submissions to encourage early bug fixing; TIRA's web front end organizes and visualizes the evaluation process.

6. *Execution Cost.* With run submissions, participants bear the costs of executing their software, since they have to bring their own hardware. With software submissions, lab organizers need to provide sufficient hardware or raise participation fees (e.g., for commercial cloud platforms). Raising fees, however, will hardly be accepted

since participants typically already own perfectly suitable hardware.

Our solution: we provide four servers, each hosting up to 20 virtual machines.

The next sub-section details our solutions.

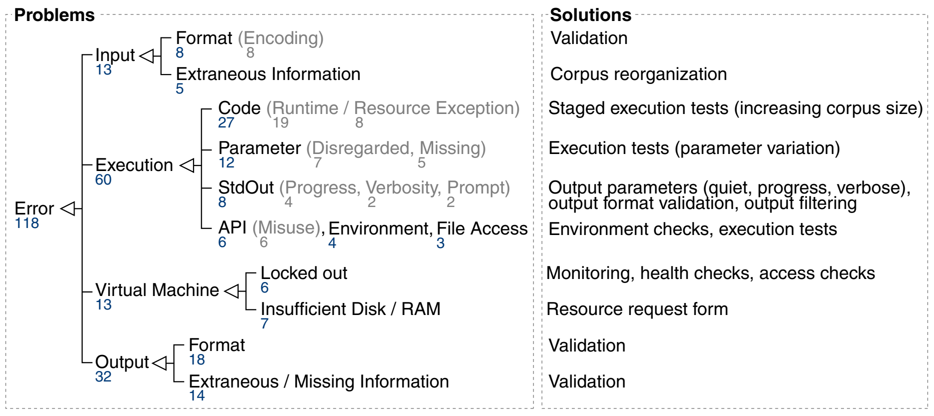
5.2 TIRA's Evaluation Toolbox

The goal of TIRA is to automate the evaluation of information retrieval experiments [17, 19]. TIRA's main capability is to integrate an experiment software into a web service and to remote control its execution. It provides a web interface to do so with the click of a button and collects and indexes results and errors for later retrieval (e.g., to construct a leaderboard or to forward error messages to the participants). Building on this basic functionality, we address items 1 to 3 as well as parts of 4 and 5 of the above list of software submission challenges by integrating virtualization, sandboxed execution, unit testing, and staged submissions.

Arbitrary execution environments as well as executing untrusted software can be safely accomplished by virtualization. Upon registration, every participant is given access to a virtual machine running at our site. Access is provided via secure shell as well as virtual network computing (i.e., remote desktop), and administrative rights are provided. This way, participants are able to set themselves up, whereas our only restriction is that their software is executable from the command line and that it has parameters for an input and output directory. To allow for a variety of environments and programming languages, two operating systems are provided, namely Microsoft Windows 7 Enterprise, and Ubuntu Linux 12.04 LTS. Although offering other operating systems would not have been a problem, none were requested.

Although misuse is unexpected, participants still have full administrative control of their virtual machines, so that it is important to take every precaution to prevent confidential data from leaking. For example, running participant software on confidential data may cause it to remain present in the virtual machine after the run is complete (e.g., within temporary files, outputs, logs, or intentionally hidden copies). Moreover, software running on the virtual machine may attempt to send copies of the data via network to an external host. To prevent such leaks, before running a software, it is moved into a so-called sandbox: (1) a snapshot of the current state of a virtual machine is taken, (2) all connections to external networks are cut, (3) the confidential test data is mounted into the virtual machine, (4) the software is run on the confidential data, (5) the output of the software is copied out of the virtual machine, and finally, (6) the virtual machine's state is reverted to that of its snapshot and all network connections are restored.

After the deployment of a software onto a virtual machine and after the participant confirms her submission, the virtual machine is moved into the sandbox. Before running the software on test data, we perform a small-scale unit test to ensure the software successfully executes. Then we run it on public data that is also accessible to participants for development purposes. The performances of this run are provided to participants so they can verify that their software behaved as expected during evaluation. Finally, the software is run on the test data and its output is checked for errors. After that, the virtual machine is moved out of the sandbox. In case of errors, participants are notified by mail and invited to re-submit a fixed version of their software.



Problems
Solutions
Error 118
Input 13
Format (Encoding) 8 8
Extraneous Information 5
Execution 60
Code (Runtime / Resource Exception) 27 19 8
Parameter (Disregarded, Missing) 12 7 5
StdOut (Progress, Verbosity, Prompt) 8 4 2 2
API (Misuse), Environment, File Access 6 6 4 3
Virtual Machine 13
Locked out 6
Insufficient Disk / RAM 7
Output 32
Format 18
Extraneous / Missing Information 14
Validation
Corpus reorganization
Staged execution tests (increasing corpus size)
Execution tests (parameter variation)
Output parameters (quiet, progress, verbose), output format validation, output filtering
Environment checks, execution tests
Monitoring, health checks, access checks
Resource request form
Validation
Validation

Figure 1. Taxonomy of 118 problems that occurred during our lab along with technical solutions that identify them automatically. The numbers indicate the amount of errors within each category.

Finally, from an organizational point of view, we found that staged submissions and engaging participants early to submit their software prototypes allows for early error correction and for getting an estimate of the final number of participants. To incentivize early software submissions, we offered an early bird submission deadline and the opportunity to get a pre-evaluation on a portion of the test data used for the final evaluation; 18 of the 46 participating teams took the opportunity to pre-evaluate their software.

5.3 Analysis of User Errors

Our current approach to error handling (item 4 of the above list) is based on a basic unit test that executes a submitted software on a very small sample of the evaluation corpus in order to learn whether it runs through. After that, the entire evaluation corpus is fed into the software. In case of errors, participants are notified by mail. In total, 1493 mails were exchanged within 392 conversations, discussing 118 errors. The number of teams experiencing at least one error is 39 from a total of 46, whereas 26 teams experienced at least two errors and one unlucky team 10. The identification of errors and the subsequent discussions induced a significant amount of manual workload. Sometimes, more than one round-trip was necessary to resolve an error. We analyzed the mails to get a better idea of what kinds of errors occurred and how they can be prevented in the future; Figure 1 organizes the errors into a taxonomy.

In general, input and output errors can be observed in traditional run submissions labs as well, whereas execution errors and virtual machine errors are exclusive to software submission labs. While the former can be easily identified or prevented by providing format validation and simplifying corpus organization, the latter require more intricate solutions or cannot be identified automatically at all. However, since half of all errors are execution errors, the work overhead for lab organizers to have them fixed can be minimized by allowing participants to perform execution tests themselves, for example, using TIRA's web front end. This way, turnaround times are minimized and no mails need be exchanged.

5.4 Evaluating Submitted Softwares Across Years

One of the primary goals of doing software submissions in a lab is to make re-evaluations of the submitted softwares on different data sets possible. Since we are doing software submissions for the second time, this forms an excellent opportunity to demonstrate this possibility by cross-evaluating software from our previous lab on the current evaluation corpora and vice versa. This way, participating in one of our labs corresponds to participating in all of them past, present, and future. Moreover, if a participant submits versions of his software in different years, this will allow to track performance improvements. We evaluated the text alignment softwares submitted to the plagiarism detection task of last year as well as those submitted this year in this way and obtained combined rankings of both years. Discussing the results here is out of scope of this section, however, they can be found in [47].

6 Conclusion and Outlook

In conclusion, the creation of standardized evaluation resources for the digital text forensics tasks plagiarism detection, author identification, and author profiling forms the basis for renewed progress to solve these problems. In this regard, our annual lab has made significant headway. With the introduction of software submissions, we hope to go even further by compiling a repository of state-of-the-art implementations of algorithms for these tasks. The research community will benefit from conducting comparative experiments against their own algorithms as well as validating new evaluation corpora by feeding them into existing softwares. More generally, we hope our lab sets a new example of how to accomplish software submissions at large, and that the TIRA experimentation platform and the tools developed for it will be adopted by other researchers. Our future research into evaluation methodology is directed at making software submissions as simple as run submissions, and to further automate the organization of evaluation labs.

Acknowledgements

This work was partially supported by the WIQ-EI IRSES project (Grant No. 269180) within the FP7 Marie Curie action.

Bibliography

[1] Y. Aleman, N. Loya, D. Vilarino Ayala, and D. Pinto. Two Methodologies Applied to the Author Profiling Task—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[2] S. Argamon and P. Juola. Overview of the International Authorship Identification Competition at PAN-2011. In *Proc. of CLEF'11*.

[3] S. Argamon, M. Koppel, J. Fine, and A. R. Shimoni. Gender, Genre, and Writing Style in Formal Written Texts. *TEXT*, 23:321–346, 2003.

[4] S. Argamon, M. Koppel, J. W.Pennebaker, and J. Schler. Automatically Profiling the Author of an Anonymous Text. *Commun. ACM*, 52(2):119–123, February 2009.

[5] T. G.Armstrong, A. Moffat, W. Webber, and J. Zobel. EvaluatIR: An Online Tool for Evaluating and Comparing IR Systems. In *Proc. of SIGIR'09*.

[6] H. Blockeel and J. Vanschoren. Experiment Databases: Towards an Improved Experimental Methodology in Machine Learning. In *Proc. of PKDD'07*, vol. 4702 of *LNCS*, 2007.
[7] J. D.Burger, J. Henderson, G. Kim, and G. Zarrella. Discriminating Gender on Twitter. In *Proc. EMNLP '11*.
[8] P. Clough and M. Stevenson. Developing a Corpus of Plagiarised Short Answers. *Lang. Resour. Eval.*, 45:5–24, March 2011.
[9] P. Clough, R. Gaizauskas, S. S. L.Piao, and Y. Wilks. METER: MEasuring TExt Reuse. In *Proc. ACL'02*
[10] D. De Roure, C. Goble, and R. Stevens. The Design and Realisation of the myExperiment Virtual Research Environment for Social Sharing of Workflows. *Future Gener. Comp. Sy.*, 25: 561–567, 2009.
[11] A. A. Caurcel Diaz and J. M. Gomez Hidalgo. Experiments with SMS Translation and Stochastic Gradient Descent in Spanish Text Author Profiling—Notebook for PAN at CLEF 2013. In Forner et al. [15].
[12] J. S. Downie. The Music Information Retrieval Evaluation Exchange (2005–2007): A Window into Music Information Retrieval Research. *Acoust. Sc. and Tech.*, 29(4):247–255, 2008.
[13] D. I. Hernandez Farias, R. Guzman-Cabrera, A. Reyes, and M. A. Rocha. Semantic-based Features for Author Profiling Identification: First Insights—Notebook for PAN at CLEF 2013. In Forner et al. [15].
[14] L. Flekova and I. Gurevych. Can We Hide in the Web? Large Scale Simultaneous Age and Gender Author Profiling in Social Media–Notebook for PAN at CLEF 2013. In Forner et al. [15].
[15] P. Forner, R. Navigli, and D. Tufis, editors. *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, 2013.
[16] L. Gillam. Readability for author profiling?—Notebook for PAN at CLEF 2013. In Forner et al. [15].
[17] T. Gollub, S. Burrows, and B. Stein. First Experiences with TIRA for Reproducible Evaluation in Information Retrieval. In *Proc. of OSIR at SIGIR'12* August 2012.
[18] T. Gollub, B. Stein, and S. Burrows. Ousting Ivory Tower Research: Towards a Web Framework for Providing Experiments as a Service. In *Proc. of SIGIR'12*.
[19] T. Gollub, B. Stein, S. Burrows, and D. Hoppe. TIRA: Configuring, Executing, and Disseminating Information Retrieval Experiments. In *Proc. of TIR at DEXA'12*. IEEE.
[20] S. Goswami, S. Sarkar, and M. Rustagi. Stylometric Analysis of Bloggers' Age and Gender. In *Proc. of ICWSM'09*.
[21] O. Haggag and S. El-Beltagy. Plagiarism Candidate Retrieval Using Selective Query Formulation and Discriminative Query Scoring—Notebook for PAN at CLEF 2013. In Forner et al. [15].
[22] J. Holmes and M. Meyerhoff. *The Handbook of Language and Gender*. Blackwell Handbooks in Linguistics. Wiley, 2003.
[23] G. Inches and F. Crestani. Overview of the International Sexual Predator Identification Competition at PAN-2012. In *Proc. of CLEF'12*.
[24] P. Juola. Authorship Attribution. *Found. and Trends in IR*, 1:234–334, 2008.
[25] P. Juola. Ad-hoc Authorship Attribution Competition. In *Proc. of ALLC'04*.
[26] P. Juola. An Overview of the Traditional Authorship Attribution Subtask. In *Proc. of CLEF'12*.
[27] M. Koppel and Y. Winter. Determining if Two Documents are by the Same Author. *Journal of the American Society for Information Science and Technology*, to appear.
[28] M. Koppel, S. Argamon, and A. R. Shimoni. Automatically Categorizing Written Texts by Author Gender. *Literary and Linguistic Computing*, 17(4):401–412, 2002.
[29] M. Koppel, J. Schler, and E. Bonchek-Dokow. Measuring Differentiability: Unmasking Pseudonymous Authors. *Journal of Machine Learning Research*, 8:1261–1276, 2007.
[30] M. Koppel, J. Schler, and S. Argamon. Authorship Attribution in the Wild. *Language Resources and Evaluation*, 45:83–94, 2011.
[31] Kong L., Qi H., Du C., Wang M., and Han Z.. Approaches for Source Retrieval and Text Alignment of Plagiarism Detection—Notebook for PAN at CLEF 2013. In Forner et al. [15].
[32] W. Y. Lim, J. Goh, and V. L. L. Thing. Content-centric age and gender profiling—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[33] A. Pastor Lopez-Monroy, M. Montes-Y-Gomez, H. Jair Escalante, L. Villasenor-Pineda, and E. Villatoro-Tello. INAOE's participation at PAN'13: Author Profiling task—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[34] M. Meina, K. Brodzinska, B. Celmer, M. Czokow, M. Patera, J. Pezacki, and M. Wilk. Ensemble-based Classification for Author Profiling using Various Features—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[35] D. Nguyen, R. Gravel, D. Trieschnigg, and T. Meder. "How Old Do You Think I Am?"; A Study of Language and Age in Twitter. *Proc. of ICWSM'13*.

[36] D. Nguyen, N. A. Smith, and C. P. Rosé. Author Age Prediction from Text Using Linear Regression. In *Proc. of LaTeCH at ACL-HLT*.

[37] B. Gopal Patra, S. Banerjee, D. Das, T. Saikh, and S. Bandyopadhyay. Automatic Author Profiling Based on Linguistic and Stylistic Features—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[38] C. Peersman, W. Daelemans, and L. Van Vaerenbergh. Predicting Age and Gender in Online Social Networks. In *Proc. of SMUC '11*.

[39] J. W. Pennebaker. *The Secret Life of Pronouns: What Our Words Say About Us*. Bloomsbury USA, 2013.

[40] J. W. Pennebaker, M. R. Mehl, and K. G. Niederhoffer. Psychological Aspects of Natural Language Use: Our Words, Our Selves. *Annual review of psychology*, 54(1):547–577, 2003.

[41] M. Potthast, B. Stein, A. Eiselt, A. Barrón-Cedeño, and P. Rosso. Overview of the 1st International Competition on Plagiarism Detection. In *Proc. of PAN at SEPLN'09*.

[42] M. Potthast, A. Barrón-Cedeño, A. Eiselt, B. Stein, and P. Rosso. Overview of the 2nd International Competition on Plagiarism Detection. In *Proc.õf CLEF'10*.

[43] M. Potthast, B. Stein, A. Barrón-Cedeño, and P. Rosso. An Evaluation Framework for Plagiarism Detection. In *Proc. of COLING'10*.

[44] M. Potthast, A. Eiselt, A. Barrón-Cedeño, B. Stein, and P. Rosso. Overview of the 3rd International Competition on Plagiarism Detection. In *Proc. of CLEF'11*.

[45] M. Potthast, T. Gollub, M. Hagen, J. Graßegger, J. Kiesel, M. Michel, A. Oberländer, M. Tippmann, A. Barrón-Cedeño, P. Gupta, P. Rosso, and B. Stein. Overview of the 4th International Competition on Plagiarism Detection. In *Proc. of CLEF'12*.

[46] M. Potthast, M. Hagen, B. Stein, J. Graßegger, M. Michel, M. Tippmann, and C. Welsch. ChatNoir: A Search Engine for the ClueWeb09 Corpus. In *Prov. of SIGIR'12*.

[47] M. Potthast, T. Gollub, M. Hagen, M. Tippmann, J. Kiesel, P. Rosso, E. Stamatatos, and B. Stein. Overview of the 5th International Competition on Plagiarism Detection. In *Proc. of CLEF'13*.

[48] M. Potthast, M. Hagen, M. Völske, and B. Stein. Crowdsourcing Interaction Logs to Understand Text Reuse from the Web. In *Proc. of ACL'13 (to appear)*. ACM, August 2013.

[49] D. A. Rodíguez Torrejón and J. M. Martín Ramos. Text Alignment Module in CoReMo 2.1 Plagiarism Detector—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[50] K. Santosh, R. Bansal, M. Shekhar, and V. Varma. Author Profiling: Predicting Age and Gender from Blogs—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[51] J. Schler, M. Koppel, S. Argamon, and J. W. Pennebaker. Effects of Age and Gender on Blogging. In *Proc. of CAAW'06*.

[52] E. Stamatatos. A Survey of Modern Authorship Attribution Methods. *Journal of the American Society for Information Science and Technology*, 60:538–556, 2009.

[53] E. Stamatatos. Plagiarism Detection Using Stopword N-grams. *Journal of the American Society for Information Science and Technology*, 62(12):2512–2527, 2011.

[54] B. Stein, S. Meyer zu Eißen, and M. Potthast. Strategies for Retrieving Plagiarized Documents. In *Proc. of SIGIR'07*

[55] Š. Suchomel, J. Kasprzak, and M. Brandejs. Diverse Queries and Feature Type Selection for Plagiarism Discovery—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[56] K. Williams, H. Chen, S. R. Chowdhury, and C. L. Giles. Unsupervised Ranking for Plagiarism Source Retrieval—Notebook for PAN at CLEF 2013. In Forner et al. [15].

[57] M. Wojnarski, S. Stawicki, and P. Wojnarowski. TunedIT.org: System for Automated Evaluation of Algorithms in Repeatable Experiments. In vol. 6086 of *LNCS*, 2010.

[58] C. Zhang and P. Zhang. Predicting Gender from Blog Posts. Technical report, University of Massachusetts Amherst, USA, 2010.