Improving the Reproducibility of PAN's Shared Tasks:
Plagiarism Detection, Author Identification, and Author Profiling

Martin Potthast,[1] Tim Gollub,[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, 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 reports on the PAN 2014 evaluation lab which hosts three shared tasks on plagiarism detection, author identification, and author profiling. To improve the reproducibility of shared tasks in general, and PAN's tasks in particular, the Webis group developed a new web service called TIRA, which facilitates software submissions. Unlike many other labs, PAN asks participants to submit running softwares instead of their run output. To deal with the organizational overhead involved in handling software submissions, the TIRA experimentation platform helps to significantly reduce the workload for both participants and organizers, whereas the submitted softwares are kept in a running state. This year, we addressed the matter of responsibility of successful execution of submitted softwares in order to put participants back in charge of executing their software at our site. In sum, 57 softwares have been submitted to our lab; together with the 58 software submissions of last year, this forms the largest collection of softwares for our three tasks to date, all of which are readily available for further analysis. The report concludes with a brief summary of each task.

1 Introduction

The term "shared task" refers to computer science events that invite researchers and practitioners to work on a specific problem of interest, *the task*.[1] The goals of a shared task may be threefold: (1) to foster the development of new theories and approaches at solving the task, (2) to implement a suited software, and (3) to evaluate the currently achievable performance. A shared task gives rise to a controlled laboratory experiment where contesting softwares are the test subjects. Within the experiment a possibly large number of problem instances of the task have to be solved, whereas the solutions of the competing softwares are compared to the true solutions. If the problem instances are representative of the population of (real-world) problem instances, the achieved performance of a software allows for judging its merits with regard to being applied in practice, as well as the validity of its underlying approach.

[1] Typical terms used in this regard are: campaign, challenge, competition, contest, or cup.

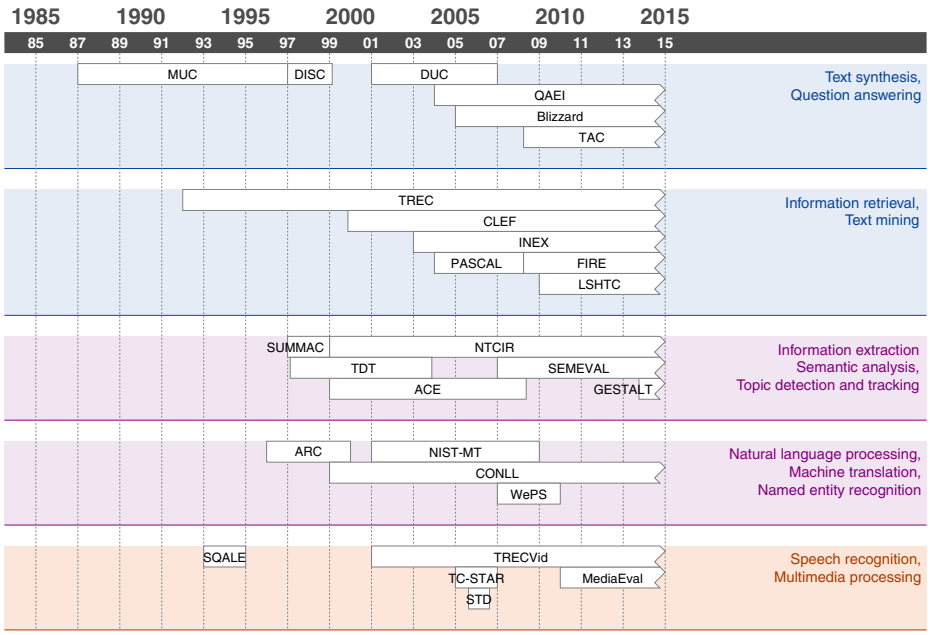

Figure 1. The last thirty years of shared tasks in the human language technologies.

Though shared tasks have a long tradition in computer science, only little is written about them. Open questions include: What are best practices to set-up a shared task? How to measure its success? What determines its success? As a step towards answering these and related questions, we have compiled an overview of well-known shared tasks in the Human Language Technologies, which is depicted in Figure 1.

1.1 Contrasting Shared Tasks by Submission Type

Our review of shared tasks in the human language technologies reveals that such tasks have been unanimously organized in the same way. Task organizers prepare a corpus comprising problem instances, where parts of the corpus are published as training data (including the ground truth) and test data (without the ground truth) respectively. Task participants develop software that solves the task based on the training data and finally run their software on the test data. Within most shared tasks, the output of this final software run (usually called a *run*, for short) is submitted to the organizers. The organizers, in turn, evaluate the submitted runs using previously announced performance measures against the ground truth of the problem instances in the test data set.

To reach higher levels of automation and reproducibility, participants may submit their executable software, this way enabling the organizers to generate runs by themselves. This approach, which we call "managed software submission," entails a lot of communication overhead and other problems, caused by the fact that now the organizing site becomes part of the software test cycle. These disadvantages are addressed by a third kind of submission type, here called "participant-in-charge software submission,"

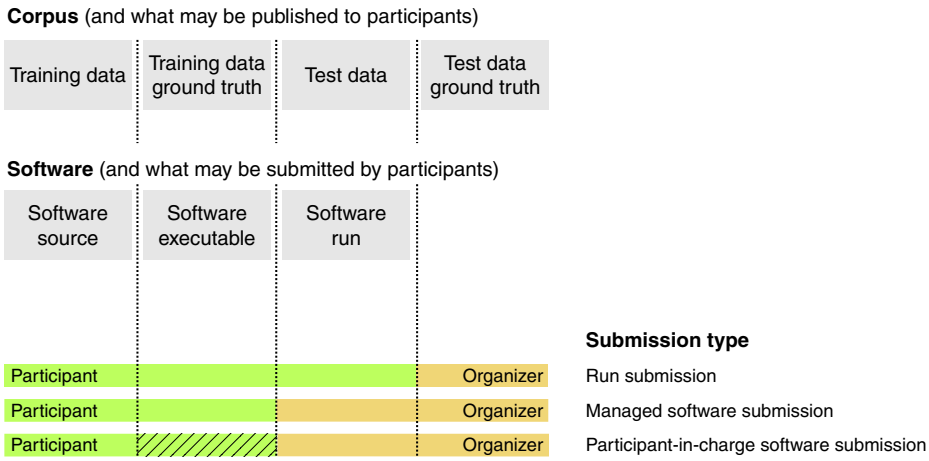

Figure 2. From top to bottom: Task organizers develop a corpus from which certain parts are published to participants. The participants in turn develop softwares from which certain parts are submitted. The extent of what is published/submitted defines the submission type: run submission, managed software submission, or participant-in-charge software submission. The last submission type enables participants to submit, execute, and optimize their softwares, using an experiment platform (such as TIRA) provided at the organizer's site, whereas the experiment platform manages a software's access to the test data set.

where a fully-fledged experiment platform is provided at the organizer's site for each participant. Though this approach is technically the most advanced, it comes along with appealing advantages: the softwares can be tested and optimized by the participants, as well as accessed, run, and archived for documentation and re-run purposes by the organizers. See Figure 2 for an illustration of the three submission types.

1.2 Related Work

The human language technologies were at the forefront of organizing shared tasks, with early initiatives dating back to the 1980's. Figure 1 places each initiative on a time line according to its primary research focus. Note that today's most established evaluation campaigns, CLEF, CONLL, INEX, NTCIR, TREC, and TRECVid, run successfully for over ten years now, each of them hosting up to dozens of specific shared tasks. The value that shared tasks provide for their respective research fields has been pointed out by Chapman *et al.* [7]. Most notably, shared tasks push the standardization of evaluation metrics and data formats, provide annotated data sets and benchmarks, foster the cooperation between academia and industry, and constitute a well defined entry point and forum for getting involved in a particular research field. The scientific impact of shared tasks has been attested by Tsikrika *et al.* [63], who analyzed the citation graph of CLEF publications. Despite their general acceptance, there are also critical voices concerning shared tasks [4,56,57]. The general argument brought forward is that shared tasks turn research fields with a great diversity of streams and ideas into a single, oversimplified task, with fixed inputs and gold-standard outputs, and a single automatic performance

metric. In addition, repeated shared tasks bear the risk that the developed approaches converge on the approach that showed most success in previous evaluations. Moreover, Potthast *et al.* [44] observe that participant do not necessary improve upon the performance of their first approach when they participate repeatedly. Given these concerns, the question is which factors influence the success or failure of shared tasks in pushing forward a research field. To the best of our knowledge, this question has not been answered, yet, within a rigorous scientific evaluation.

1.3 Contributions

This paper reports on the latest results of our efforts to improve the reproducibility of shared tasks in general, and that of PAN's three shared tasks in particular, namely plagiarism detection, author identification, and author profiling. We introduce the TIRA experimentation platform as a web service for shared tasks: it implements a participant-in-charge software submission platform that hands the responsibility of successful software execution back to participants. This way, inviting software submissions for a shared task becomes significantly less cumbersome, and, given further development, it may reduce the work overhead to a point at which inviting software submission may become as straightforward as inviting run submissions has been previously.

All of the above has not been developed haphazardly, but the development process was tightly integrated with PAN over the past years, using our lab as a beta testing platform for our developments. While first plans for TIRA have been discussed long ago, at PAN 2012 we first invited managed software submissions for one of PAN's shared tasks. Based on this experience, developments commenced which allowed us to scale managed software submissions to all three of PAN's recurring shared tasks in 2013, whereas this year marks the introduction of participant-in-charge software submissions based on the TIRA web service. This way, we can not only claim to have developed the first participant-in-charge software submission platform, but also that this platform is battle-tested based on handling three demanding shared tasks with more than 100 software submissions in total since 2012.

2 TIRA: A Web Service for Shared Tasks

This section reports on our efforts to minimize the organizational overhead of software submissions and the ongoing development of the TIRA experimentation platform [14,15]. For three years in a row, our lab has invited software submissions, and for the second time, this was done for all shared tasks. This year, 57 softwares have been submitted to our three tasks all of which were handled using TIRA. In previous work we identified challenges that handling software submissions at scale entail [13]: (1) development environment diversity, (2) untrusted software execution, (3) data leakage, (4) error handling, (5) execution responsibility, and (6) execution cost. Until last year, the first three challenges have been our primary concern, while the focus of this year is on the two challenges of error handling and execution responsibility. Our long-term goal is to make inviting software submissions for shared tasks as simple as inviting run submissions, avoiding the deficiencies of the latter while adding the benefits of the

former. All of these efforts are consolidated by developing TIRA's evaluation tools that facilitate software submissions and shared tasks in general. For the first time, we provide public access to these tools via a new web front end.[2]

2.1 Software Submissions: Who is Responsible for their Successful Execution?

A major obstacle to a widespread adoption of managed software submissions in shared tasks is the shift of responsibility for a successful software execution. Submitted software is not necessarily free of errors—even more, experience shows that the majority of the participants submit their software prematurely, yet, being convinced from its flawlessness. This fact lets organizers unwillingly become part of the debugging process of each participant's software, whereas the turnaround time to find and fix errors increases severely, especially when both parties are not working simultaneously (i.e., reside in different time zones). Failure on the part of organizers to run a submitted software, to check its output for errors of any kind (e.g., not every execution error results in a crash), and to give participants feedback in a timely manner may cause participants to miss submission deadlines. The risk of this happening is increased by the fact that many participants start working only just in time before a deadline, so that organizers have to handle all submissions at the same time. Besides, prolonged back-and-forth between participants and organizers cause by software errors bears a high potential for friction. As a result, organizers may come to the conclusion they have little to gain but trouble, whereas the benefits of software submissions, such as reproducibility, may be considered insufficient payback.

In previous years, we experienced the following with managed software submissions [13]: to get the 58 softwares submitted in 2013 running for evaluation, 1493 mails had to be exchanged in order to fix runtime errors. It must be noted, though, that we were working hand-in-hand with participants, and that, surprisingly, most participants were not at all disgruntled by having to revisit their software over and over again to fix errors. While our previous versions of TIRA have helped us to manage software submissions in an organized manner, our goal now is to put participants back in charge of their own software (see Figure 2). Therefore, we develop user interfaces for TIRA, which allow participants to remotely control software execution and to collect runtime feedback, thus eliminating the need for organizers to intervene in fixing software execution errors. Figure 3 illustrates the interfaces provided to both parties.

In what follows, both the user interfaces and the workflow of participants and organizers to complete a shared task are described in detail.

2.2 Life of a Participant

From the perspective of a participant (Alice, in the following), a software submission via TIRA happens within three basic steps: first, deployment of the software to a given virtual machine, second, configuration of the software for remote execution, and third, remote execution of the software on the available training and test data. The interfaces on the left side of Figure 3 are used for this purpose, whereas the latter two steps are

[2] http://www.tira.io

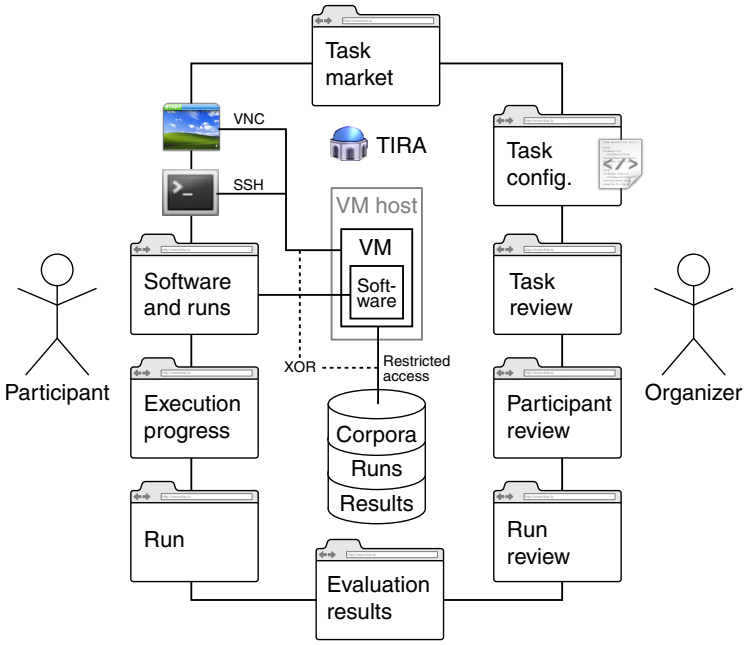

Figure 3. TIRA's interfaces for participants (left), organizers (right), and the public (top, bottom).

accomplished via TIRA's new web interface. The web interface marks an important step forward in terms of putting Alice in charge of deploying her software: it ensures that Alice neither gains direct access to the test data nor to the ground truth of a shared task, but it still allows her to evaluate her software and to obtain filtered runtime feedback. In this regard, TIRA serves as a remote control for evaluation.

TIRA encapsulates Alice's software in a virtual machine that is set up once she registers for a shared task. As depicted in Figure 3, Alice has two ways to access her virtual machine, namely a remote desktop connection and an SSH connection. Alice retains full administrative rights inside her virtual machine, so that she can set up her preferred development environment and deploy her software. To prevent misuse, virtual machines are not allowed to communicate with each other, and, their outgoing bandwidth is limited. By default, virtual machines have only restricted access to TIRA's database, so that only the training data of each task can be read. Once a software has been successfully deployed and tested manually, participants use TIRA's web interface to complete the second and third step outlined above.

For each participant of a shared task, TIRA serves a remote control page for the respective virtual machine, the deployed software, and the software runs. After signing in with her account for the first time, Alice can configure the execution details of her software. Figure 4 shows Alice's software control page in a state after completed configuration and a few successfully executed runs. The software control page is divided into four panels:

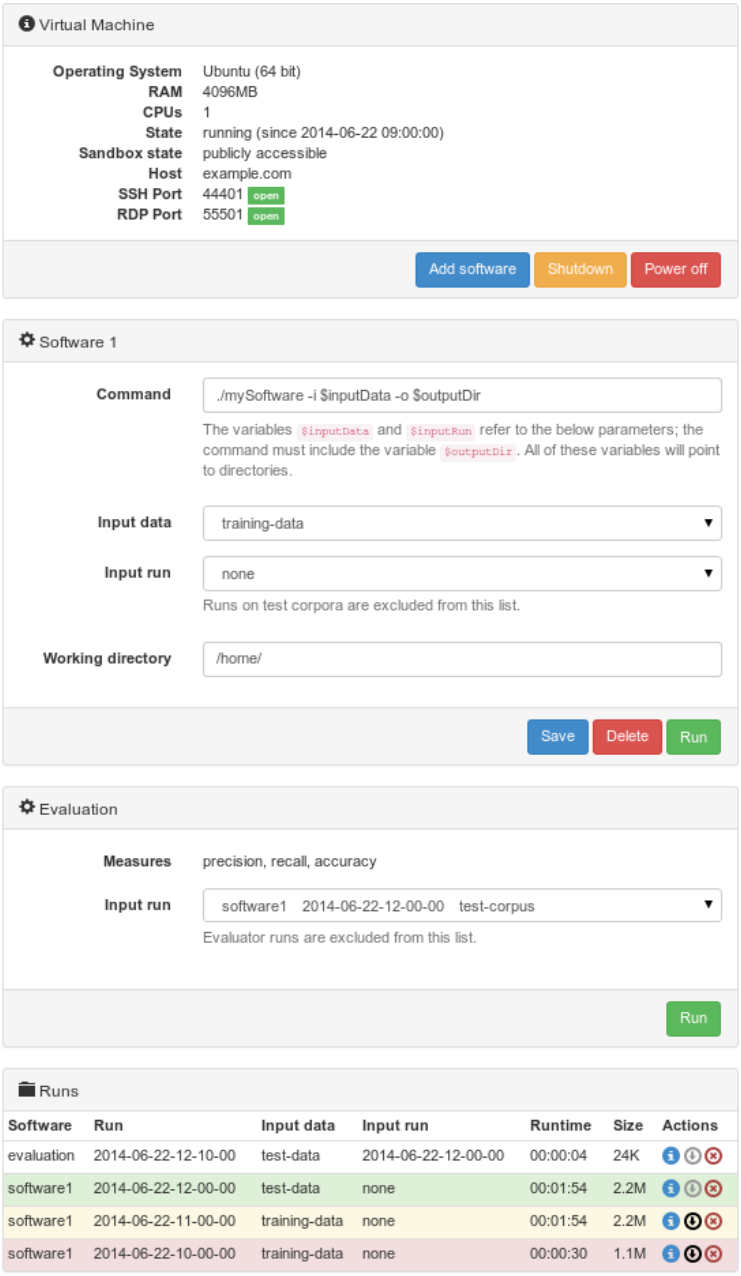

Figure 4. TIRA's web interface for participants to remote control the execution of their software and to review their runs for a given shared task.

Virtual Machine Overview of the virtual machine, including information about the operating system, RAM, CPUs, its running state, VM host, and connectivity. The virtual machine can be turned off at the click of a button either by sending a shutdown signal to the operating system, or by powering it off. Clicking "Add Software" creates a new software panel. Alice may deploy an arbitrary number of softwares for the shared task onto her virtual machine, e.g., to compare different paradigms or variants of an approach at solving the task. Each software can be configured individually on the software control page.

Software 1 Configuration of a software that has been previously deployed on the virtual machine. The software must be executable as a POSIX-conform command line. Mandatory parameters can be defined by organizers of the shared task. In this case, they include variables for input data and the output directory, and optionally for an input run (i.e., a previous run of one of Alice's softwares). If necessary, the working directory in which the program shall be executed can be specified. Alice may adjust an existing software configuration and save its state, she may delete it, or she may proceed to execute the software. Note that if Alice deletes a software it is not actually deleted on the server, but only hidden from view; rationale for this is to allow organizers to reconstruct Alice's actions for reasons of cheating prevention. The runs obtained from running a software are listed in the "Runs" panel.

Evaluation Run an evaluation software on a given run. This is a special type of software provided by task organizers which processes an input run and outputs the results of the task's performance measures. Once Alice has finished her first successful run on a given input data, she uses this panel to evaluate it. The runs obtained from an evaluation software are also listed in the "Runs" panel.

Runs List of runs that have been obtained either from running a software or from running an evaluation. The table lists run details including software, timestamp (which also serves as run ID), input data, input run, runtime, size on disk, and further actions that can be taken. The colorization indicates a run's status with regard to being successful, where red indicates severe errors, yellow indicates warnings, green indicates complete success, and white indicates that the run has not yet been reviewed. Runs are checked automatically for validity with the shared task's expected output format, and they may be reviewed manually by organizers. Actions that can be taken on each run include viewing more details (the blue *i*-icon), downloading it (the black arrow down), and deleting it (the red x). It is here where Alice first encounters the limitations that TIRA imposes for runs on test data sets: all test data sets are by default hidden from participants, which is why all possible communication channels about test data must be filtered or closed as well. Therefore, TIRA prevents Alice from downloading runs on test data sets (the download action shown grayed is inactive) to foreclose that a malicious software outputs the data itself instead of output that is valid for a given shared task.

The software control page does not display all of the aforementioned panels immediately, but only after Alice has completed the necessary steps. At first, it only shows the virtual machine panel; then, once Alice clicks on "Add Software", a software panel appears; and finally, once Alice runs her software for the first time, the evaluation panel and the runs panel are added after the run is completed. While a software is running, the

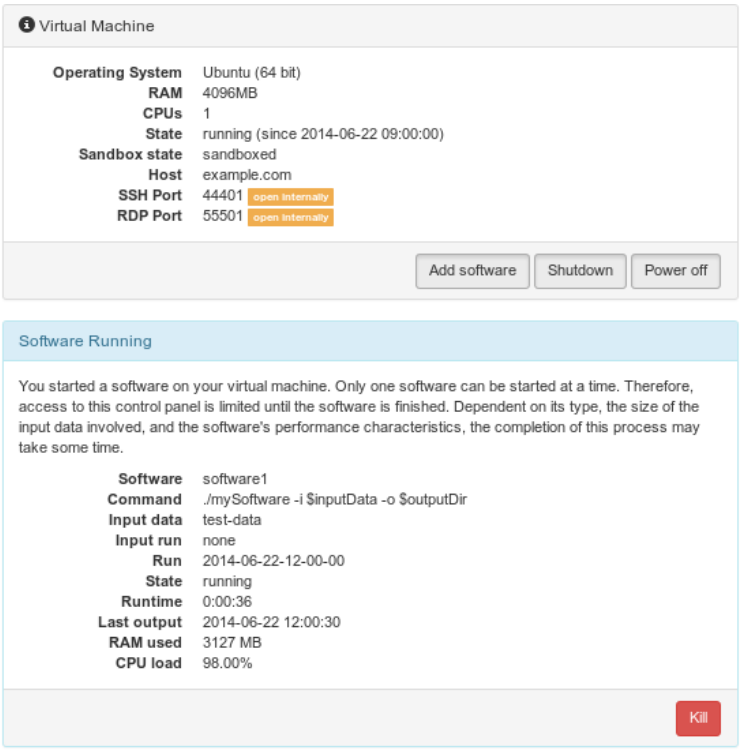

Figure 5. TIRA's web interface to monitor the progress of a running software.

software control page is replaced with the software progress monitoring page which is divided into two panels, as exemplified in Figure 5:

Virtual Machine Just as on the software control page, the virtual machine panel shows the current state of Alice's virtual machine while the software is running. Before a software is started, the virtual machine is moved into a so-called sandbox: the machine is disconnected from the Internet so that no outside connections are possible, a snapshot is taken to save the machine's state before the software is executed, and, the input data is mounted read-only into the virtual machine as a shared folder. This sandbox state is indicated to Alice in the corresponding list entry as well as by the connectivity flags. Only if a machine has been successfully moved into the sandbox, the software is executed. While a software is running, the buttons to add a software configuration panel as well as those to shutdown or power off the virtual machine are deactivated so that the running software is not interrupted accidentally. After the software terminates, the output is stored in TIRA's database as a run, and the virtual machine is automatically moved out of the sandbox: the input data is unmounted, the virtual machine is restored to the state of the snapshot that was taken just before it was moved into the sandbox, and then it is reconnected to the Internet. Restoring the virtual machine to the snapshot taken ensures that no information about the input data remains in the virtual machine, be it in cache, in

temporary files, or in purposefully hidden files. Disconnecting the virtual machine from the Internet while a software is executed ensures that no data can be sent to an unauthorized third party.

Software Running Overview of a running software, including the software's ID, the executed command, the parameters, the run ID, and the running state. Moreover, the current runtime, the time of the last write access to the output directory, the currently used RAM, and the CPU load are displayed and updated periodically. This way, Alice has a way of making sure her software is still working. If, for any reason, Alice wishes to kill her software before it terminates by itself, she may click on the "Kill" button. Before the software is killed, its output up to this point is stored in TIRA's data base as an incomplete run for later inspection.

After her run has completed and the virtual machine has been moved out of the sandbox, Alice's browser shows the software control page again as in Figure 4. The new run appears in the runs table. To make sure the run was successful, Alice clicks on the *i*-icon which redirects her to a run details page for the run in question, as shown in Figure 6a. The details shown about a run are as follows:

Overview Details about the run, including the software that was used, the run ID, parameters, whether the run can be downloaded, runtime details, its size, and the numbers of lines, files, and directories found. Whether the run can be downloaded depends on whether the input data was a test data set or not. As outlined above, runs on test data sets, by default, cannot be downloaded to foreclose data leakage. Besides the runtime, more in-depth runtime details are given, so that Alice can judge whether her software made good use of the hardware available to the virtual machine. For example, if she finds there are many page faults or even swaps, this indicates the software uses too much memory. The size and numbers of lines, files, and directories provide quantitative feedback to quickly verify output sanity, whereas it depends on the task which of these values is most illuminating.

Review Review of this run provided by both automatic validation and organizers. In Alice's case, an organizer reviewed the displayed run and found that it does not contain any obvious errors. In case of errors, explanations are displayed here that give insight into their nature and severity.

Stdout Standard output stream (stdout) which was recorded when executing the software. If Alice's software outputs information to stdout, it will be displayed here. However, in the case of runs on test data sets, the amount of information that is displayed can be limited. In the example, the limit is the 100 last chars of the stdout text. This limitation shall prevent Alice from outputting problem instances to stdout in order to inspect them. This communication channel can be closed entirely on a per-data set basis, for example, if confidential data has to be handled.

Stderr Standard error output stream (stderr) which was recorded when executing the software. While nothing was recorded in the example, the same filtering is applied as for the stdout stream.

File List Directory tree which displays file names and their sizes found in the run. Alice may use this information to determine whether her run has output all the files and directories that are expected, and whether their names and organization are correct.

(a) Details page of a software run.

(b) Excerpt of the details page of an evaluation run.

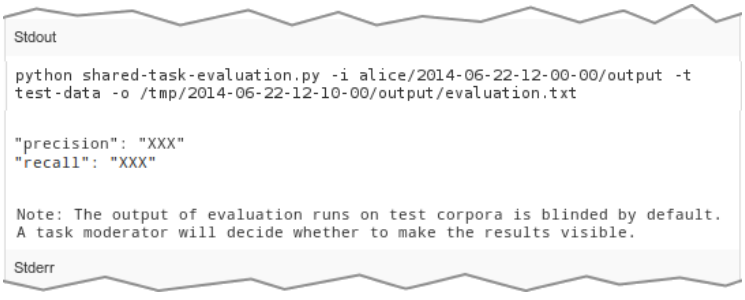

Figure 6. TIRA's web interfaces for participants to review runs.

The run details page shall provide Alice with the information necessary to determine whether her remote software execution was successful. Unless the software has been executed on a test data set, Alice may also download the run for local inspection. If she is satisfied with the run, she may proceed to evaluate it using the evaluation software. The resulting evaluation can again be inspected just like before, whereas the corresponding run details page lists the information pertaining to the evaluation software's run when receiving Alice's software run as input. Figure 6b shows an excerpt of an evaluation run details page that Alice will see. The evaluation software typically prints the evaluation results directly to stdout, however, in case the evaluated software run was on a test data set, the results are blinded by default (i.e., the performance values are replaced by "XXX"). Our rationale for blinding the evaluation results is twofold: (1) participants of shared tasks are not supposed to see their software's performances before the task organizers decide to publish them, and, (2) participants are not supposed to optimize their software against the test data, for example, by means of trial and error. This way, the decision of when, if, and how the evaluation results of a given shared task are released is at the full discretion of its organizers. Moreover, just as with filtering stdout and stderr output, the organizers may adjust blinding on a per-data set basis.

After completing her evaluation run, Alice is done; she has submitted her software to the virtual machine, made sure it works to the specifications of the shared task by running it on the available data sets and inspecting the runs for errors, and finally executed the evaluation software on her previous software runs. While Alice can now relax, it is time for the organizers of the shared task to get busy.

2.3 Life of an Organizer

From the perspective of an organizer (Bob, for example), using TIRA to manage software submissions for a shared task can be done in three simple steps: first, configuration of the shared task in TIRA; second, supervision of participant progress; and third, compilation and publication of the task's evaluation results. The interfaces on the right side of Figure 3 are used for this purpose. The configuration of a shared task in TIRA is done in a text-based configuration file. Configurable aspects include the data sets and their privacy settings as outlined in the previous section, the evaluation softwares, the command line parameters required for submitted softwares, and various messages displayed on task-specific web pages. The web interface for task configuration basically displays the configuration file as is and allows for editing it; we omit a screenshot for brevity.

In terms of supervising his shared task while it is underway, Bob has three interfaces at his disposal, an overview of participants who have started to work on the shared task, an overview of runs of each participant, and the run details of each participant's runs:

Task Participants (Figure 7a) Overview of participants who have configured at least one software for Bob's shared task on their software control page, including their user name, signed in status, numbers of softwares that are configured, deleted, and running, and, numbers of runs that are finished, reviewed, and unreviewed. These figures give Bob an idea of whether the participants of his task are actively engaged, but it also hints problems that may require Bob's attention. The number of deleted

(a) Overview of a task's participants.

Participants in *Shared Task*

User	Signed in	Softwares	Deleted	Now Running	Runs	Reviewed	Unreviewed	Actions
Alice	yes	7	6	none	63	62	**1**	
Carol	no	1	0	6 days, 8:37:25	4	3	**1**	
Dan	no	1	0	none	5	0	**5**	
Eve	no	3	1	none	16	16	**0**	
Frank	no	3	0	none	56	56	**0**	
Mallory	no	1	0	none	4	0	**4**	
Oscar	no	1	0	none	4	0	**4**	
Peggy	no	1	0	none	4	0	**4**	
Sybil	no	3	2	none	5	5	**0**	
Trent	no	1	0	none	4	0	**4**	

(b) Overview of a participant's runs.

Runs of *Alice* on *test-corpus*

Software	Run	Input run	Size	Lines	Files	Dirs	Review	Actions
evaluation	2014-06-22-12-10-00	2014-06-22-12-00-00	24K	36	1	0	**todo**	
software1	2014-06-22-12-00-00	none	2.2M	5180	518	0	done	
software1	2014-06-22-11-00-00	none	2.2M	5180	518	0	done	
software1	2014-06-22-10-00-00	none	1.1M	2590	259	0	done	
software1	2014-06-22-09-00-00[DEL]	none	0.55M	1290	129	0	done	
software1	2014-06-22-08-00-00[DEL]	none	1K	20	2	0	done	

Figure 7. TIRA's web interfaces for organizers to review a task's participants.

softwares may indicate that a participant has trouble setting herself up. In the case of Alice, six of seven softwares have been deleted, so that it may be the case that Alice had some trouble getting the software configuration right. In the case of Carol, Bob observes that her software has been running for more than six days straight, which may be an indication that the software is not working as anticipated, given that the expected runtime of a software for Bob's shared task is a lot lower. Bob may contact the respective participants and offer his help. Moreover, the number of unreviewed runs indicates that some runs have not yet been checked for errors by an organizer. To do so, Bob clicks on the review action (the blue eye-icon in the Actions column) to review all of Alice's runs; he is redirected to the participant runs page described next.

Participant Runs (Figure 7b) Overview of a participant's runs on a per-data set basis, including the software that was used, run ID, input run, size, numbers of lines, files, and directories, and whether a run has been reviewed. The colorization indicates a run's status with regard to being successful, where red indicates severe errors, yellow indicates warnings, green indicates complete success, and white indicates that the run has not yet been reviewed. Unlike in the runs table on Alice's software control page, this table shows figures which relate to judging a run's success by checking its size or the numbers of lines, files, or directories against the expectation

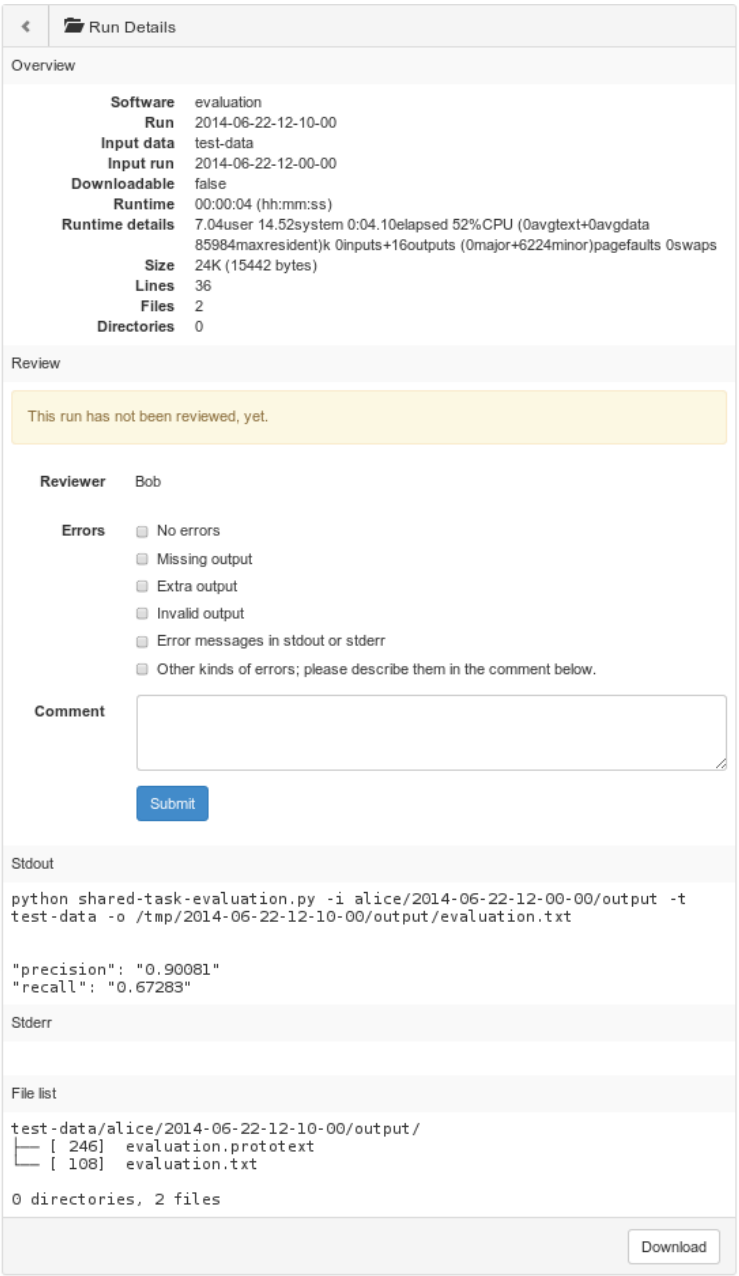

Figure 8. TIRA's web interfaces for organizers to review runs.

for a given data set. Unlike Alice, Bob has access to all of Alice's runs including those that have been deleted by Alice. The deleted runs are annotated with the superscript "DEL." Moreover, since Bob is task organizer, there are no restrictions with regard to downloading runs. To review the outstanding unreviewed run, Bob clicks again on the corresponding review action and is redirected to the run details page described next.

Run details (Figure 8) The run details page corresponds to that which Alice can access. It displays the same information about the run, but there are four differences. (1) it offers a review form in which Bob can enter his review, (2) the standard output streams are not filtered, (3) the output of evaluation softwares is not blinded, and (4) the button to download the run is always activated. Based on the complete information about the run, Bob can easily review it, which usually takes only a couple of seconds. Bob's review consists of checking for common errors, such as missing output, extra output, output validity, as well as error messages that have been printed to either standard output stream. These are the common errors that have been observed to occur frequently in previous years [13], whereas Bob has the opportunity to write a short comment about uncommon errors he observes. Bob can supply run verification software for his task that checks runs automatically, however, at least for runs that will be used for the final evaluation results of a shared task, a quick review should be done to foreclose unforeseen errors. This reduces Bob's responsibility for the successful evaluation of Alice's software to a level similar to shared tasks that invite run submissions.

The supervision duties of task organizers cannot be entirely avoided. In shared tasks that invite run submission, the organizers usually do not have to intervene until after the submission deadline. Only then, they learn how many participants actually submit a run and how many of the submitted runs are valid as to the specifications of the shared task. In the extreme case, it is only after the run submission deadline, when actual examples of runs on test data sets are available, that the organizers realize that parts of the data set or the run formats are unfit for their evaluation goals. With software submissions based on TIRA, these risks can be minimized since organizers have a chance to observe early bird participants and make adjustments as the shared task progresses. An added benefit of supervising a shared task using TIRA is that organizers learn early on how many participants actually work toward making a submission to the task, whereas with run submissions, the success or failure of a shared task in terms of number of participants will only become apparent after the run submission deadline. If Bob were to observe that only few participants start using TIRA, he may react by engaging with those who registered but did not start, yet, or by advertising the task some more in the community.

Once the submission deadline passed, and all participants successfully evaluated their runs on the test data sets of Bob's shared task, he proceeds to reviewing the performances and publishing the results. For this purpose, TIRA has an overview of all evaluation runs on a per-data set basis (see Figure 9a):

Evaluations Results Overview of evaluation runs and the performance results obtained, including user name, software, ID of the evaluation run, ID of the software run that served as input to the evaluation run, and performance values, dependent

(a) Overview of a task's evaluation results for organizers.

Evaluations on *test-corpus*

User	Software	Evaluation	Input run	Precision	Recall	Actions
Alice	software1	2014-06-22-12-10-00	2014-06-22-12-00-00	0.90081	0.67283	
Carol	software3	2014-06-15-17-38-08	2014-06-15-17-35-38	0.85744	0.29661	
Dan	software2DEL	2014-06-16-17-17-21	2014-06-16-16-54-38DEL	0.96022	0.84248	
Dan	software3	2014-06-23-20-43-59	2014-06-23-20-17-48	0.96007	0.84511	
Dan	software1	2014-06-16-18-03-43	2014-06-16-17-21-44	0.96243	0.83473	
Eve	software1	2014-06-01-12-52-02	2014-06-21-05-56-23	0.82882	0.84156	
Frank	software10	2014-06-23-13-31-42	2014-06-23-13-24-21	0.92522	0.81819	
Mallory	software1	2014-06-20-23-28-21	2014-06-17-09-28-40	0.87171	0.91539	
Oscar	software1	2014-06-19-00-54-42	2014-06-18-23-50-04	0.92757	0.88916	
Peggy	software3	2014-06-22-03-36-34	2014-06-22-03-33-32	0.90032	0.80267	
Sybil	software2	2014-06-22-02-56-09	2014-06-22-02-49-41	0.90770	0.79931	
Sybil	software4	2014-06-22-16-55-56	2014-06-22-16-49-05	0.89179	0.80590	
Trent	software5	2014-06-15-16-24-05	2014-06-15-15-53-28	0.86606	0.91984	

(b) Overview of a task's *published* evaluation results.

Evaluations on *test-corpus*

User	Precision	Recall	Runtime
Alice	0.90081	0.67283	00:04:17
Carol	0.85744	0.29661	00:00:56
Dan	0.96007	0.84511	00:19:32
Eve	0.82882	0.84156	00:05:18
Frank	0.92522	0.81819	00:02:49
Mallory	0.87171	0.91539	00:05:37
Oscar	0.92757	0.88916	00:57:15
Peggy	0.90032	0.80267	00:00:31
Trent	0.86606	0.91984	00:22:10

Figure 9. TIRA's web interfaces for a task's evaluation results.

on the measures computed by a given evaluation software. The colorization of the table cells for both run IDs corresponds to that of the run reviews mentioned above. This helps Bob to decide which are successful evaluations. All evaluation runs of all participants on a given data set are listed, including deleted runs. For example, there are multiple runs for participant Dan and Sybil. Bob gets to decide which of their runs are going to be published; there are a number of reasonable decision rules in this situation: (1) all of them (2) the chronologically first or last successful run, (3) the run chosen by the respective participant, or (4) the best performing run according to a given performance measure. While the decision rule that is applied can be chosen by Bob, it is currently not enforced automatically. In the Actions column, there are two publishing options, namely publication of evaluation results to the public evaluation results page (the globe icon), and publication of evaluation results to the respective participant (the person icon). As can be seen in the exam-

ple, Bob has already globally published evaluations runs for all but one participant. Two of Dan's runs are published only to him, and for Sybil's two runs Bob still needs to make a decision.

The published runs appear on a public evaluation results page that can be found on TIRA alongside each shared task. Figure 9b shows the performance values of the evaluations that Bob decided to publish for his shared task. While he proceeds to announce the results to participants as well as to the scientific community, this is not necessarily the end of the story.

Shared tasks are organized for a reason, and that reason is not to host an individual run-once competition, but to foster research around a problem of interest. While shared tasks are sometimes organized repeatedly, at some point, they are discontinued, whereas later on there are still researchers who want to compare their approach to those of the task's participants. Based on TIRA, this will be easily possible long after a shared task is over, since all the evaluation resources required to run an evaluation are hosted and kept in running state. Moreover, if new evaluation corpora appear, all previously developed approaches can be re-evaluated on the new corpora, since they are also kept in running state inside their virtual machines. This way, TIRA paves the way for ongoing, "asynchronous" evaluations around a shared task while ensuring that everyone is evaluated using the exact same environment. That is, of course, as long as TIRA prevails.

In what follows, we report on the results of three shared tasks which have been organized using TIRA, and for which a total of 57 softwares have been submitted this year. The tasks are plagiarism detection, author identification, and author profiling.

3 Plagiarism Detection

This section summarizes the evaluation of 17 plagiarism detectors that have been submitted to our corresponding shared tasks. A complete version of our report can be found in [45], where a more in-depth analysis of the obtained results as well as a survey of detection approaches is given. We evaluate different aspects of plagiarism and text reuse detectors within the two tasks source retrieval and text alignment. Both have been identified as integral parts of plagiarism detection [61]. Since we have organized plagiarism detection-related tasks for six years in a row, we observe a recurrent multi-year life cycle, which can be divided into three phases, namely an innovation phase, a consolidation phase, and a production phase. In the innovation phase, new evaluation resources are being developed; in the consolidation phase, based on the feedback and results obtained from the innovation phase, the new evaluation resources are developed to maturity; and in the production phase, the task is repeated with little changes to allow participants to build upon what has been accomplished, and, to make the most of the prior investment in developing the new evaluation resources. Meanwhile, new ideas are being developed to introduce further innovation. Both, the source retrieval task and the text alignment task are now in production. In what follows, we briefly overview related work as well as the evaluation setup and the results obtained for both tasks.

3.1 Related Work

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 [49].[3] During the first three labs, a total of 43 plagiarism detectors have been evaluated using this framework [50,41,42]. 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 [12]. 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 three labs [43,44,45]. Besides the mentioned corpora, there are two other ones that comprise text reuse, namely the Meter corpus [8] and the Clough09 corpus [9]. 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.

3.2 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 For the evaluation of source retrieval from the web, we consider the real-world scenario of an author who uses a web search engine to retrieve documents in order to reuse text from them. A plagiarism detector typically uses a search engine, too, to find reused sources of a given document. Over the past years, we assembled the

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

Table 1. Source retrieval results with respect to retrieval performance and cost-effectiveness.

Team (alphabetical order)	Downloaded Sources			Total Workload		Workload to 1st Detection		No Detect.	Runtime
	F_1	$prec$	rec	Queries	Dwlds	Queries	Dwlds		
Elizalde	0.34	0.40	0.39	54.5	33.2	16.4	3.9	7	**04:02:00**
Kong	0.12	0.08	0.48	83.5	207.1	85.7	24.9	6	24:03:31
Prakash	0.39	0.38	**0.51**	60.0	38.8	8.1	3.8	7	19:47:45
Suchomel	0.11	0.08	0.40	**19.5**	237.3	**3.1**	38.6	**2**	45:42:06
Williams	**0.47**	**0.57**	0.48	117.1	**14.4**	18.8	**2.3**	4	39:44:11
Zubarev	0.45	0.54	0.45	37.0	18.6	5.4	**2.3**	3	40:42:18

necessary building blocks to allow for a meaningful evaluation of source retrieval algorithms. The setup was described in much more detail in last year's task overview [44]. The main components are two associated search engines for the ClueWeb corpus 2009 (ClueWeb09).[4] This corpus represents one of the most widely adopted web crawls and it is regularly used for large-scale web search-related evaluations. It consists of about one billion web pages, half of which are English ones. Indri[5] and ChatNoir [46] are currently the only publicly available search engines that index the ClueWeb09 corpus. For developer convenience, we also provide a proxy server which unifies the APIs of the search engines. At the same time, the proxy server logs all accesses to the search engines for later analysis.

Evaluation Corpus The evaluation corpus employed for source retrieval is based on the Webis text reuse corpus 2012 (Webis-TRC-2012) [48,47]. The corpus consists of 297 documents that have been written by 27 writers who worked with our setup: given a topic, a writer used ChatNoir to search for source material on that topic while preparing a document of 5700 words length on average, reusing text from the found sources. In the last years, we sampled 98 documents from the Webis-TRC-2012 as training and test documents. This year, these documents were provided for training, and another 99 documents were sampled as test documents. The remainder of the corpus will be used within future instances of this task.

Evaluation Results Table 1 shows the performances of the six plagiarism detectors that implemented source retrieval. Their 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 its usage costs. Moreover, we measure precision and recall of downloaded documents with regard to the true source documents and compute F_1. For lack of a 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.

None of the detectors dominates the others in terms of all of the employed measures, whereas three detectors share the top scores among them. The detector of Williams *et al.* [68] achieves the best trade-off between precision and recall in terms of F_1 as well

[4] http://lemurproject.org/clueweb09

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

Table 2. Text alignment performances of the 2014 participants on the 2013 test data.

Team	PlagDet	Recall	Precision	Granularity	Runtime
Sanchez-Perez	**0.87818**	**0.87904**	0.88168	1.00344	00:25:35
Oberreuter	0.86933	0.85779	0.88595	1.00369	00:05:31
Palkovskii	0.86806	0.82637	0.92227	1.00580	01:10:04
Glinos	0.85930	0.79331	**0.96253**	1.01695	00:23:13
Shrestha	0.84404	0.83782	0.85906	1.00701	69:51:15
R. Torrejón	0.82952	0.76903	0.90427	1.00278	**00:00:42**
Gross	0.82642	0.76622	0.93272	1.02514	00:03:00
Kong	0.82161	0.80746	0.84006	1.00309	00:05:26
Abnar	0.67220	0.61163	0.77330	1.02245	01:27:00
Alvi	0.65954	0.55068	0.93375	1.07111	00:04:57
Baseline	0.42191	0.34223	0.92939	1.27473	00:30:30
Gillam	0.28302	0.16840	0.88630	**1.00000**	00:00:55

as best precision, whereas the detector of Prakash and Saha [51] achieves best recall. Suchomel and Brandejs [62]'s detector requires least query workload, least queries until first detection, and detects source documents for almost all of the test documents. The detector of Williams *et al.* [68], however, performs worst in terms of total querying workload, since it requires 117 queries on average. Posing a query to a search engine may entail significant costs, whereas downloading a document is considered much less costly. By comparison, the detector of Zubarev and Sochenkov [70] achieves a similarly good trade-off between precision and recall with much less querying costs and comparable downloading costs. This detector also competes in terms of workload until first true positive detection with less than 6 queries and about 2 downloads on average.

3.3 Text Alignment

In text alignment, given a pair of documents, the task is to identify all contiguous passages of reused text between them. This task has a long tradition at PAN, yet, every year new ideas emerge at solving this task. Since this task is in its production phase, we have made little changes compared to last year in order to allow participants to optimize against the existing evaluation resources.

Evaluation Corpus As an evaluation corpus we reused both the training and test data from last year [44]. Reusing existing evaluation resources bears the risk that participants may overfit their approaches against them, thereby diminishing the generalizability of their respective approaches. This is why we opted not to tell participants the fact that we reuse last years training and test data, and, we generated a small supplemental evaluation corpus to which participants had no prior access. The supplemental evaluation corpus has been constructed in the same way as last years test corpus to allow for comparability of results. However, only a subset of last year's strategies to obfuscate the reused text passages of a plagiarism cases have been employed. Therefore, last years test data serve as reference for evaluation. Last years corpus consists of 5185 pairs of documents, which contain reused passages of text of varying lengths and obfuscation, such as paraphrasing, random text modification, cyclic translation, and summarization.

Moreover, there is verbatim reuse to simulate naive plagiarist behavior. The supplemental test corpus contains 4800 pairs of documents with a limited set of obfuscations, namely verbatim copies and random text modifications.

Evaluation Results Table 2 shows the overall performance of eleven plagiarism detectors that implemented text alignment. The detailed performances of each detector with regard to different kinds of obfuscation can be found in [45]. Performances are measured using precision and recall at character level as well as granularity (i.e., how often the same plagiarism case is detected). Based on these values we compute the PlagDet score by dividing F_1 by the granularity's logarithm. The detectors are ranked by PlagDet.

The best performing detector is that of Sanchez-Perez *et al.* [54]; a new contender in this task, closely followed by the detectors of Oberreuter and Eiselt [36] and Palkovsii and Belov [37]. The latter have also been evaluated in previous years, whereas the detector of Palkovskii and Belov [37] has significantly improved. Over the years, it can be observed that the differences in performance between detectors are getting smaller and smaller, which may indicate that improving the algorithms that solve this task becomes more difficult.

4 Author Identification

This section summarizes the evaluation of 13 author identifiers that have been submitted to our corresponding shared task. A complete version of our report can be found in [59], where a more in-depth analysis of the obtained results as well as a survey of detection approaches is given. Author identification is the most prevalent field of authorship analysis in terms of published studies [21,58]. The problem variant "authorship attribution" can be viewed as a closed-set classification task where all possible candidate authors (the classes) are known. This is typical for forensic applications, where, based on certain restrictions such as access to specific material or knowledge of specific facts, the investigators of a case can provide a set of suspects. A more general definition of the authorship attribution problem leads to an open-set classification task, where the true author of a disputed text is not necessarily among the set of candidate authors. Compared to the closed-set attribution scenario, this setting is much more difficult, especially if the size of the candidate author set is small [24]. Finally, if the set of candidate authors is singleton, we get the author *verification* problem, which is a fundamental problem in authorship attribution since any problem setting can be decomposed into a series of verification problems [26].

4.1 Related Work

Previous work on author verification has been evaluated using sample texts in one language only (Greek [60], Dutch [17,30], English [25,26]) and a specific genre (newspaper articles [60], student essays [30], fiction [25], newswire stories [19], poems [19], blogs [26]). Author verification was also included in previous editions of PAN: the author identification task at PAN-2011 included three author verification problems [1],

PAN-2013 focused on author verification and provided corpora in English, Greek, and Spanish [22]. However, the size of the corpora was small and covered only one genre per language.

A variety of performance measures have been used in previous work on this task including false acceptance and false rejection rates [60,17], accuracy [25,26], recall, precision, F_1 [30], balanced error rate [19], recall-precision graphs [26] macro-average precision and recall [1], and ROC graphs [22]. Unfortunately, these measures are not able to explicitly estimate the ability of an approach to leave problems unanswered—a fact which is crucial in a cost-sensitive task like this.

The author identification task at PAN-2013 successfully introduced software submissions, this way enabling reproducibility of the results and future evaluation on different corpora. A meta-model combining all the submitted methods achieved the best overall performance, showing the potential of heterogeneous models in this task [22].

4.2 Evaluation Setup

Similar to PAN-2013 [22], PAN-2014 focuses on author verification. Given both a set of known documents written by the same author and a single questioned document, the task is to determine whether or not the questioned document was written by the author of the other documents. Each verification problem has been carefully configured to ensure that all known and the questioned documents are matched for genre, register, theme, and the date of writing. The number of known documents has been limited to be at most five, while a variety of languages and genres is covered. The document lengths vary from a few hundred to a few thousand words, depending on the genre.

The participants were asked to submit a software that takes the document language and genre as input parameters. For each verification problem they had to provide a score from the interval [0,1], corresponding to the probability of a positive answer (i.e., the known and the questioned documents are by the same author). To label a verification problem as unanswered, a probability score of 0.5 could be assigned.

4.3 Evaluation Corpus

The PAN-2014 corpus comprises author verification problems in the four languages Dutch, English, Greek, and Spanish. For Dutch and English there are two genres in separate parts of the corpus. Beyond language and genre there is a variety of known texts per problem and text length. The training and evaluation sets are balanced in the number of positive and negative examples. The corpus size is significantly larger than the corresponding corpus of PAN-2013.

The Dutch corpus part is a transformed version of the CLiPS Stylometry Investigation (CSI) corpus [64]. This recently released corpus contains documents of the two genres essay and review. All documents are written by language students, native Dutch speakers, at the University of Antwerp between 2012 and 2014.

The English essays are derived from a corpus of English-as-second-language students, the Uppsala Student English (USE) corpus [3], which was originally intended to become a tool for research on foreign language learning. It consists of university-level

full-time students' essays. Taking advantage of the USE corpus meta-information, we defined two main constraints: (1) each document in the collection, known or questioned, must contain at least 500 words, and, (2) the number of known documents in a case must range between one and five. We took also advantage of meta information to define case-generation rules that deal with matching terms and student age. The outlined measures allowed us for creating cases where the authors share a similar background. Finally, a source USE document could be considered at most twice: once in a positive case and once in a negative case.

The set of English novels are our attempt to provide a narrower focus in terms of both content and writing style than existing collections. Instead of simply focusing on a single genre or time period, the texts focus on a very small subgenre of speculative and horror fiction, known as the "Cthulhu Mythos." It is based on the writings of the American H. P. Lovecraft ("Lovecraftian horror"), a shared universe with a theme of human ineffectiveness when facing powerful "cosmic horrors." It is characterized by extremely florid prose and an unusual vocabulary. Perhaps most significantly, many of the elements of this genre are unusual terms, thus creating a shared element that is unusual in normal English prose. Similarly, the overall theme and tone of these stories is strongly negative. The documents cover an extended length of time, from Lovecraft's original work to modern fan-fiction. The documents were collected from a variety of on-line sources including the Project Gutenberg[6] and FanFiction.[7]

The Greek part of the corpus comprises newspaper opinion articles published in the Greek weekly newspaper TO BHMA between 1996 and 2012.[8] The length of each article is at least 1,000 words, while the number of known texts per problem varies between one and five. For each verification problem, we ensured strong thematic similarity, indicated by the occurrence of certain keywords. In contrast to PAN-2013, there was no stylistic analysis of the texts to identify authors with similar styles or texts of the same author. The Spanish part of the corpus refers to the same genre and is built from opinion articles of the Spanish newspaper El-Pais.[9] Again, the formed author verification problems ensure thematic similarities between the articles.

4.4 Performance Measures

The probability scores provided by the participants are used to built ROC curves, whereas the area under curve, AUC, is used as a scalar evaluation measure [10]. In addition, the performance measures for this task are able to account for unanswered problems: if there is much uncertainty about a decision, it is possible to leave the problem unanswered. We adopted the c@1 measure, originally proposed for question answering tasks, which extends the accuracy based on the number of unanswered problems [38]. The measure rewards participants who maintain a large number of correct answers of high confidence. To rank the participants, a final score is defined as the product of AUC and c@1. In addition, the efficiency of the submitted methods is measured in terms of the elapsed runtime.

[6] http://www.gutenberg.org
[7] http://www.fanfiction.net
[8] http://www.tovima.gr
[9] http://www.elpais.com

Table 3. Author identification results in terms of final score (AUC*c@1) and runtime.

Team	Overall	Essays		Articles		Novels	Reviews	Runtime
		en	nl	es	gr	en	nl	(hh:mm:ss)
Meta classifier	0.566	0.531	0.867	0.709	0.635	0.472	0.428	
Khonji	**0.490**	0.349	0.770	**0.698**	**0.720**	0.458	0.479	20:59:40
Frery	0.484	**0.513**	0.821	0.581	0.436	0.360	0.347	00:06:42
Castillo	0.461	0.318	0.741	0.558	0.501	0.386	0.247	03:59:04
Moreau	0.451	0.372	0.755	0.634	0.565	0.313	0.375	01:07:34
Mayor	0.450	0.318	**0.823**	0.539	0.621	0.407	0.299	05:26:17
Zamani	0.426	0.322	0.525	0.468	0.470	0.476	0.362	02:37:25
Satyam	0.400	0.459	0.489	0.248	0.356	0.380	**0.525**	02:52:37
Modaresi	0.375	0.350	0.378	0.416	0.294	**0.508**	0.247	**00:00:38**
Jankowska	0.367	0.284	0.732	0.586	0.497	0.225	0.357	07:38:18
Halvani	0.335	0.338	0.399	0.423	0.367	0.293	0.316	00:00:54
Baseline	0.325	0.288	0.685	0.378	0.452	0.202	0.322	00:21:10
Vartapetiance	0.308	0.270	0.517	0.436	0.281	0.245	0.260	01:07:39
Layton	0.306	0.363	0.307	0.299	0.403	0.260	0.261	27:00:01
Harvey	0.304	0.312	0.396	0.514	0.000	0.283	0.170	01:06:19

4.5 Evaluation Results

We received 13 submissions of research teams from Australia, Canada (2), France, Germany (2), India, Iran, Ireland, Mexico (2), United Arab Emirates, and United Kingdom. The participants submitted and evaluated their author verification software within the TIRA framework [13]. A separate run for each corpus part (combination of language and genre) was performed.

As a challenging baseline for the submitted approaches a language-independent author verification method from PAN-2013 [20] was employed: the winner of the competition in terms of AUC; note that the respective approach has not been trained on the PAN-2014 corpus. Moreover, following the practice of PAN-2013 [22], we examined the performance of a meta-model that averages the answers of all submitted systems.

The evaluation results in terms of the final score (AUC $\cdot$ c@1), the baseline method, and the meta-classifier are shown in Table 3. The overall (micro-averaged) performances along with the total runtime are also given. In terms of the average performance of all submitted approaches, the Dutch essays are the easiest problems, while the Dutch reviews are the hardest. The latter can be partially explained by the fact that only one known document per problem is used and the review texts are very short. Note that there is a different winner for each corpus part, with the exception of the overall winner approach by Khonji and Iraqi, who won on both the Greek and Spanish corpus subsets. In general, the majority of the submitted methods outperformed the baseline, while the performance of the meta-classifier is significantly better than any individual method.

Similar to PAN-2013, the overall winner was a modification of the *Impostors* method [26]. The performance of this approach was notably stable on all six corpus subsets. This demonstrates the potential of extrinsic verification methods, which transform author verification from a one-class classification task towards a binary classification task, using additional texts from other authors as negative examples. In addition, the significantly larger training set allowed participants to explore, for the first time,

the use of eager learning methods. Such an approach, followed by the second overall winner, can be effective as well as efficient.

5 Author Profiling

This section summarizes the evaluation of 10 author profilers that have been submitted to our corresponding shared task. A complete version of our report can be found in [52], where a more in-depth analysis of the obtained results as well as a survey of detection approaches is given. Author profiling tries to determine an author's gender, age, native language, personality type, etc. solely by analyzing an author's texts.

5.1 Related Work

The study of how certain linguistic features vary according to the profile of their authors is a subject of interest for several different areas such as psychology, linguistics and, more recently, computational linguistics. 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.* [2] 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 [18,5]. With the rise of the social media, Koppel *et al.* [23] 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.* [55] 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.* [16] 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.* [39] compiled a dataset for the purpose of gender and age prediction from Netlog.[10] Studying short texts, Zhang and Zhang [69] experimented with segments of blog posts and obtained 72.1% accuracy for gender prediction. Similarly, Nguyen *et al.* [34] studied the use of language and age among Dutch Twitter users. They modeled age as a continuous variable (as they had previously done in [35]), 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. Our lab was the first to offer author profiling as a shared task. At PAN 2013 [53] we aimed at identifying age and gender from a large corpus collected from social media. Most of the participants used combinations of style-based features such as frequency of punctuation marks, capital letters, quotations, and so on, together

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

Table 4. Joint identification results in terms of accuracy for Author Profiling.

Team	Overall	Social Media		Blogs		Twitter		Reviews
		en	es	en	es	en	es	en
López-Monroy	**0.2895**	0.1902	0.2809	**0.3077**	**0.3214**	**0.3571**	0.3444	0.2247
Liau	0.2802	0.1952	**0.3357**	0.2692	0.2321	0.3506	0.3222	**0.2564**
Shrestha	0.2760	**0.2062**	0.2845	0.2308	0.2500	0.3052	**0.4333**	0.2223
Weren	0.2349	0.1914	0.2792	0.2949	0.1786	0.2013	0.2778	0.2211
Villena-Román	0.2315	0.1905	0.1961	**0.3077**	0.2321	0.2078	0.2667	0.2199
Marquardt	0.1998	0.1428	0.2102	0.1282	0.2679	0.1948	0.3111	0.1437
Baker	0.1677	0.1277	0.1678	0.1282	0.2321	0.1688	0.2111	0.1382
Baseline	0.1404	0.0930	0.1820	0.0897	0.0536	0.1494	0.2333	0.1821
Mechti	0.1067	0.1244	0.1060	0.0897	0.1786	0.0584	0.1444	0.0451
Castillo Juarez	0.0946	0.1445	0.1254	0.1795	0.0893	–	–	0.1236
Ashok	0.0834	0.1318	–	0.1282	–	0.1948	–	0.1291

with POS tags and content-based features such as Latent Semantic Analysis, bag-of-words, $tf \cdot idf$, dictionary-based words, topic-based words, and so on. Notably, the winner of the PAN 2013 task [29] used second order representations based on relationships between documents and profiles, whereas another well-performing approach is the use of collocations of the winner of the English task [33].

5.2 Evaluation Corpora

In the Author Profiling task at PAN 2013 [53] participants approached the task of identifying age and gender in a large corpus collected from social media. At PAN 2014, we continue to study the gender and age aspects of the author profiling problem, however, four data sets of different genres were considered—social media, blogs, Twitter, and hotel reviews—both in English and Spanish. We annotated age with the following classes: 18-24; 25-34; 35-49; 50-64; and 65+.

The social media data set was built by sampling parts of the PAN 2013 evaluation corpus. We selected only authors with an average number of words greater than 100 in their posts. We also reviewed manually the data in order to remove authors who appear to be fake profiles such as bots. The blogs and Twitter data sets were manually collected and annotated by three annotators. The Twitter data set was built in collaboration with RepLab,[11] where the main goal of author profiling in the context of reputation management on Twitter is to decide how influential a given user is in a domain of interest. For each blog, we provided up to 25 posts and for each twitter profile, we provided up to 1000 tweets. The hotel review data set is derived from another corpus that was originally used for aspect-level rating prediction [66].[12] The original corpus was crawled from the hotel review site TripAdvisor[13] and manually checked for quality and compliance with the format requirements of PAN 2014.

[11] http://nlp.uned.es/replab2014

[12] http://times.cs.uiuc.edu/~wang296/data

[13] http://www.tripadvisor.com

5.3 Evaluation Results

In Table 4 joint identification accuracies for both gender and age prediction are shown per data set and averaged over all data sets, which also serves as ranking criterion. The approach of López-Monroy *et al.* [28] performs best overall. Moreover, it can be seen that (1) the highest joint accuracies were achieved on Twitter data, and, (2) the smallest joint accuracies were achieved in English social media and hotel reviews. It is an open question why these differences can be observed, whereas possible explanations may be that people express themselves more spontaneously on Twitter compared to the other genres, whereas the low scores are due to the approaches' difficulty of predicting gender in social media and age in hotel reviews.

In summary, simple content features, such as bag-of-words or word n-grams achieve best accuracies. Bag-of-words features are used by Liau and Vrizlynn [27], word n-grams are used by Maharjan *et al.* [31], and term vector models are used by Villena-Román and González-Cristóbal [65]. They achieved competitive performances on almost all data sets. Notably, Weren *et al.* [67] employ information retrieval features and Marquardt *et al.* [32] mix content and style features.

6 Conclusion and Outlook

In conclusion, the creation of the TIRA evaluation platform has fundamentally changed the way we organize shared tasks at PAN. While our initial goal was to improve the reproducibility of our shared tasks, the technology that was developed as a result of this endeavor is applicable for more than just software submissions. For example, an initial analysis of the access logs that we recorded allows for a heretofore unknown, exciting insight into the research in progress of participants of a shared tasks. Specific usage patterns can be discerned in real-time which allow organizers of a shared task to engage with participants who exert usage patterns related to software execution errors. Moreover, the overall participation in a shared task can be monitored as it happens, whereas today, most organizers will only learn if their task was successful right after the run submission deadline, when it becomes clear how many of the registered participants actually submit a run.

Besides the exciting opportunities that arise from TIRA, all of these benefits are now readily available to PAN's three tasks plagiarism detection, author identification, and author profiling. For these tasks, we have already assembled an archive of more than 100 virtual machines on which the state of the art approaches are deployed in a manner that makes them immediately executable. It is still unforeseeable how this will impact future research in these tasks.

Acknowledgements

We thank the organizing committees of PAN's shared tasks Walter Daelemans, Patrick Juola, Miguel Angel Sánchez Pérez, Ben Verhoeven, Alberto Barrón-Cedeño, and Irina Chugur. Moreover, we thank our student assistants Anna Beyer and Matthias Busse for helping with maintaining TIRA. Our special thanks go to all of PAN's participants. This work was partially supported by the WIQ-EI IRSES project (Grant No. 269180) within the FP7 Marie Curie action.

References

1. Argamon, S., Juola, P.: Overview of the International Authorship Identification Competition at PAN-2011. In: Petras, V., Forner, P., Clough, P. (eds.) Working Notes Papers of the CLEF 2011 Evaluation Labs (Sep 2011), http://www.clef-initiative.eu/publication/working-notes
2. Argamon, S., Koppel, M., Fine, J., Shimoni, A.R.: Gender, Genre, and Writing Style in Formal Written Texts. TEXT 23, 321–346 (2003)
3. Axelsson, M.: USE–The Uppsala Student English Corpus: An Instrument for Needs Analysis. ICAME Journal 24, 155–157 (2000), http://nora.hd.uib.no/icame/ij24/
4. Belz, A.: Shared-task Evaluations in HLT: Lessons for NLG. In: Proceedings of INLG-2006 (2006)
5. Burger, J.D., Henderson, J., Kim, G., Zarrella, G.: Discriminating Gender On Twitter. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 1301–1309. EMNLP '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011)
6. Cappellato, L., Ferro, N., Halvey, M., Kraaij, W. (eds.): CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers, 15-18 September, Sheffield, UK. CEUR Workshop Proceedings, CEUR-WS.org (2014), http://www.clef-initiative.eu/publication/working-notes
7. Chapman, W.W., Nadkarni, P.M., Hirschman, L., D'Avolio, L.W., Savova, G.K., Uzuner, O.: Overcoming Barriers To NLP For Clinical Text: The Role Of Shared Tasks And The Need For Additional Creative Solutions. Journal of the American Medical Informatics Association : JAMIA 18(5), 540–543 (Sep 2011), http://dx.doi.org/10.1136/amiajnl-2011-000465
8. Clough, P., Gaizauskas, R., Piao, S., Wilks, Y.: METER: MEasuring TExt Reuse. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. pp. 152–159. ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA (2002)
9. Clough, P., Stevenson, M.: Developing a Corpus of Plagiarised Short Answers. Lang. Resour. Eval. 45, 5–24 (March 2011)
10. Fawcett, T.: An Introduction to ROC Analysis. Pattern Recognition Letters 27(8), 861–874 (2006)
11. Forner, P., Navigli, R., Tufis, D. (eds.): CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers, 23-26 September, Valencia, Spain (2013), http://www.clef-initiative.eu/publication/working-notes
12. Gollub, T., Hagen, M., Michel, M., Stein, B.: From Keywords to Keyqueries: Content Descriptors for the Web. In: Gurrin, C., Jones, G., Kelly, D., Kruschwitz, U., de Rijke, M., Sakai, T., Sheridan, P. (eds.) 36th International ACM Conference on Research and Development in Information Retrieval (SIGIR 13). pp. 981–984. ACM (Jul 2013), http://dl.acm.org/citation.cfm?id=2484181
13. Gollub, T., Potthast, M., Beyer, A., Busse, M., Rangel, F., Rosso, P., Stamatatos, E., Stein, B.: Recent Trends in Digital Text Forensics and its Evaluation. In: Forner, P., Müller, H., Paredes, R., Rosso, P., Stein, B. (eds.) Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 4th International Conference of the CLEF Initiative (CLEF 13). pp. 282–302. Springer, Berlin Heidelberg New York (Sep 2013)
14. Gollub, T., Stein, B., Burrows, S.: Ousting Ivory Tower Research: Towards a Web Framework for Providing Experiments as a Service. In: Hersh, B., Callan, J., Maarek, Y., Sanderson, M. (eds.) 35th International ACM Conference on Research and Development in Information Retrieval (SIGIR 12). pp. 1125–1126. ACM (Aug 2012)

15. Gollub, T., Stein, B., Burrows, S., Hoppe, D.: TIRA: Configuring, Executing, and Disseminating Information Retrieval Experiments. In: Tjoa, A.M., Liddle, S., Schewe, K.D., Zhou, X. (eds.) 9th International Workshop on Text-based Information Retrieval (TIR 12) at DEXA. pp. 151–155. IEEE, Los Alamitos, California (Sep 2012)
16. Goswami, S., Sarkar, S., Rustagi, M.: Stylometric Analysis of Bloggers' Age and Gender. In: Adar, E., Hurst, M., Finin, T., Glance, N.S., Nicolov, N., Tseng, B.L. (eds.) ICWSM. The AAAI Press (2009)
17. van Halteren, H.: Linguistic Profiling for Author Recognition and Verification. In: Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics. ACL '04, Association for Computational Linguistics, Stroudsburg, PA, USA (2004), http://dx.doi.org/10.3115/1218955.1218981
18. Holmes, J., Meyerhoff, M.: The Handbook of Language and Gender. Blackwell Handbooks in Linguistics, Wiley (2003)
19. Jair-Escalante, H., Montes-y Gómez, M., Villasenor-Pineda, L.: Particle Swarm Model Selection For Authorship Verification. In: Proceedingsof the 14th Iberoamerican Conference on Pattern Recognition. pp. 563–570 (2009)
20. Jankowska, M., Keselj, V., Milios, E.: CNG Text Classification for Authorship Profiling Task—Notebook for PAN at CLEF 2013. In: Forner et al. [11]
21. Juola, P.: Authorship Attribution. Foundations and Trends in Information Retrieval 1, 234–334 (2008)
22. Juola, P., Stamatatos, E.: Overview of the Author Identification Task at PAN-2013. In: P., T.D.E.F. (ed.) Notebook Papers of CLEF 2013 LABs and Workshops (CLEF-2013) (2013)
23. Koppel, M., Argamon, S., Shimoni, A.R.: Automatically Categorizing Written Texts by Author Gender (2003)
24. Koppel, M., Schler, J., Argamon, S.: Authorship Attribution in the Wild. Language Resources and Evaluation 45, 83–94 (2011)
25. Koppel, M., Schler, J., Bonchek-Dokow, E.: Measuring Differentiability: Unmasking Pseudonymous Authors. J. Mach. Learn. Res. 8, 1261–1276 (Dec 2007), http://dl.acm.org/citation.cfm?id=1314498.1314541
26. Koppel, M., Winter, Y.: Determining if Two Documents are Written by the Same Author. Journal of the American Society for Information Science and Technology 65(1), 178–187 (2014)
27. Liau, Y., Vrizlynn, L.: Submission to the Author Profiling Competition at PAN-2014. http://www.webis.de/research/events/pan-14 (2014), From the Institute for Infocomm Research, Singapore
28. López-Monroy, A.P., Montes-y Gómez, M., Jair-Escalante, H., Villasenor-Pineda, L.: Using Intra-Profile Information for Author Profiling—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
29. López-Monroy, A.P., Montes-y-Gómez, M., Jair-Escalante, H., Villasenor-Pineda, L., Villatoro-Tello, E.: INAOE's Participation at PAN'13: Author Profiling task—Notebook for PAN at CLEF 2013. In: Forner et al. [11]
30. Luyckx, K., Daelemans, W.: Authorship Attribution and Verification with many Authors and Limited Data. In: Proceedings of the Twenty-Second International Conference on Computational Linguistics (COLING 2008). pp. 513–520. Coling 2008 Organizing Committee, Manchester, UK (2008)
31. Maharjan, S., Shrestha, P., Solorio, T.: A Simple Approach to Author Profiling in MapReduce—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
32. Marquardt, J., Fanardi, G., Vasudevan, G., Moens, M.F., Davalos, S., Teredesai, A., Cock, M.D.: Age and Gender Identification in Social Media—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]

33. Meina, M., Brodzinska, K., Celmer, B., Czokow, M., Patera, M., Pezacki, J., Wilk, M.: Ensemble-based Classification for Author Profiling Using Various Features—Notebook for PAN at CLEF 2013. In: Forner et al. [11]
34. Nguyen, D., Gravel, R., Trieschnigg, D., Meder, T.: "How old do you think I am?"; A study of Language and Age in Twitter. Proceedings of the Seventh International AAAI Conference on Weblogs and Social Media (2013)
35. Nguyen, D., Smith, N.A., Rosé, C.P.: Author Age Prediction from Text Using Linear Regression. In: Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities. pp. 115–123. LaTeCH '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011)
36. Oberreuter, G., Eiselt, A.: Submission to the 6th International Competition on Plagiarism Detection. http://www.webis.de/research/events/pan-14 (2014), From Innovand.io, Chile
37. Palkovskii, Y., Belov, A.: Developing High-Resolution Universal Multi-Type N-Gram Plagiarism Detector—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
38. Peñas, A., Rodrigo, A.: A Simple Measure to Assess Non-Response. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1. pp. 1415–1424. HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA (2011), http://dl.acm.org/citation.cfm?id=2002472.2002646
39. Peersman, C., Daelemans, W., Vaerenbergh, L.V.: Predicting Age and Gender in Online Social Networks. In: Proceedings of the 3rd international workshop on Search and mining user-generated contents. pp. 37–44. SMUC '11, ACM, New York, NY, USA (2011)
40. Pennebaker, J.W., Mehl, M.R., Niederhoffer, K.G.: Psychological Aspects of Natural Language Use: Our Words, Our Selves. Annual review of psychology 54(1), 547–577 (2003)
41. Potthast, M., Barrón-Cedeño, A., Eiselt, A., Stein, B., Rosso, P.: Overview of the 2nd International Competition on Plagiarism Detection. In: Braschler, M., Harman, D., Pianta, E. (eds.) Working Notes Papers of the CLEF 2010 Evaluation Labs (Sep 2010), http://www.clef-initiative.eu/publication/working-notes
42. Potthast, M., Eiselt, A., Barrón-Cedeño, A., Stein, B., Rosso, P.: Overview of the 3rd International Competition on Plagiarism Detection. In: Petras, V., Forner, P., Clough, P. (eds.) Working Notes Papers of the CLEF 2011 Evaluation Labs (Sep 2011), http://www.clef-initiative.eu/publication/working-notes
43. Potthast, M., Gollub, T., Hagen, M., Graßegger, J., Kiesel, J., Michel, M., Oberländer, A., Tippmann, M., Barrón-Cedeño, A., Gupta, P., Rosso, P., Stein, B.: Overview of the 4th International Competition on Plagiarism Detection. In: Forner, P., Karlgren, J., Womser-Hacker, C. (eds.) Working Notes Papers of the CLEF 2012 Evaluation Labs (Sep 2012), http://www.clef-initiative.eu/publication/working-notes
44. Potthast, M., Gollub, T., Hagen, M., Tippmann, M., Kiesel, J., Rosso, P., Stamatatos, E., Stein, B.: Overview of the 5th International Competition on Plagiarism Detection. In: Forner, P., Navigli, R., Tufis, D. (eds.) Working Notes Papers of the CLEF 2013 Evaluation Labs (Sep 2013), http://www.clef-initiative.eu/publication/working-notes
45. Potthast, M., Hagen, M., Beyer, A., Busse, M., Tippmann, M., Rosso, P., Stein, B.: Overview of the 6th International Competition on Plagiarism Detection. In: Cappellato, L., Ferro, N., Halvey, M., Kraaij, W. (eds.) CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2014), http://www.clef-initiative.eu/publication/working-notes
46. Potthast, M., Hagen, M., Stein, B., Graßegger, J., Michel, M., Tippmann, M., Welsch, C.: ChatNoir: A Search Engine for the ClueWeb09 Corpus. In: Hersh, B., Callan, J., Maarek, Y., Sanderson, M. (eds.) 35th International ACM Conference on Research and Development in Information Retrieval (SIGIR 12). p. 1004. ACM (Aug 2012)

47. Potthast, M., Hagen, M., Völske, M., Stein, B.: Crowdsourcing Interaction Logs to Understand Text Reuse from the Web. In: Fung, P., Poesio, M. (eds.) Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 13). pp. 1212–1221. ACL (Aug 2013), http://www.aclweb.org/anthology/P13-1119
48. Potthast, M., Hagen, M., Völske, M., Stein, B.: Exploratory Search Missions for TREC Topics. In: Wilson, M.L., Russell-Rose, T., Larsen, B., Hansen, P., Norling, K. (eds.) 3rd European Workshop on Human-Computer Interaction and Information Retrieval (EuroHCIR 2013). pp. 11–14. CEUR-WS.org (Aug 2013), http://www.cs.nott.ac.uk/ mlw/euroHCIR2013/proceedings/paper3.pdf
49. Potthast, M., Stein, B., Barrón-Cedeño, A., Rosso, P.: An Evaluation Framework for Plagiarism Detection. In: Huang, C.R., Jurafsky, D. (eds.) 23rd International Conference on Computational Linguistics (COLING 10). pp. 997–1005. Association for Computational Linguistics, Stroudsburg, Pennsylvania (Aug 2010)
50. Potthast, M., Stein, B., Eiselt, A., Barrón-Cedeño, A., Rosso, P.: Overview of the 1st International Competition on Plagiarism Detection. In: Stein, B., Rosso, P., Stamatatos, E., Koppel, M., Agirre, E. (eds.) SEPLN 09 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09). pp. 1–9. CEUR-WS.org (Sep 2009), http://ceur-ws.org/Vol-502
51. Prakash, A., Saha, S.: Experiments on Document Chunking and Query Formation for Plagiarism Source Retrieval—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
52. Rangel, F., Rosso, P., Chugur, I., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daelemans, W.: Overview of the Author Profiling Task at PAN 2014. In: Cappellato, L., Ferro, N., Halvey, M., Kraaij, W. (eds.) CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2014), http://www.clef-initiative.eu/publication/working-notes
53. Rangel, F., Rosso, P., Koppel, M., Stamatatos, E., Inches, G.: Overview of the Author Profiling Task at PAN 2013—Notebook for PAN at CLEF 2013. In: Forner et al. [11]
54. Sanchez-Perez, M., Sidorov, G., Gelbukh, A.: A Winning Approach to Text Alignment for Text Reuse Detection at PAN 2014—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
55. Schler, J., Koppel, M., Argamon, S., Pennebaker, J.W.: Effects of Age and Gender on Blogging. In: AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs. pp. 199–205. AAAI (2006)
56. Scott, D., Moore, J.: An NLG Evaluation Competition? Eight reasons to be Cautious. In: Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation. pp. 22–23 (2007)
57. Smeaton, A.F., Over, P., Kraaij, W.: Evaluation Campaigns and TRECvid. In: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval. pp. 321–330. MIR '06, ACM, New York, NY, USA (2006), http://doi.acm.org/10.1145/1178677.1178722
58. Stamatatos, E.: A Survey of Modern Authorship Attribution Methods. Journal of the American Society for Information Science and Technology 60, 538–556 (2009)
59. Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P., Sanchez-Perez, M., Barrón-Cedeño, A.: Overview of the Author Identification Task at PAN 2014. In: Cappellato, L., Ferro, N., Halvey, M., Kraaij, W. (eds.) CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2014 (to appear)), http://www.clef-initiative.eu/publication/working-notes
60. Stamatatos, E., Fakotakis, N., Kokkinakis, G.: Automatic Text Categorization in Terms of Genre and Author. Comput. Linguist. 26(4), 471–495 (Dec 2000), http://dx.doi.org/10.1162/089120100750105920

61. Stein, B., Meyer zu Eißen, S., Potthast, M.: Strategies for Retrieving Plagiarized Documents. In: Clarke, C., Fuhr, N., Kando, N., Kraaij, W., de Vries, A. (eds.) 30th International ACM Conference on Research and Development in Information Retrieval (SIGIR 07). pp. 825–826. ACM, New York (Jul 2007)
62. Suchomel, Šimon., Brandejs, M.: Heterogeneous Queries for Synoptic and Phrasal Search—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
63. Tsikrika, T., de Herrera, A.G.S., Müller, H.: Assessing the Scholarly Impact of ImageCLEF. In: Proceedings of the Second International Conference on Multilingual and Multimodal Information Access Evaluation. pp. 95–106. CLEF'11, Springer-Verlag, Berlin, Heidelberg (2011), http://dl.acm.org/citation.cfm?id=2045274.2045290
64. Verhoeven, B., Daelemans, W.: Clips Stylometry Investigation (CSI) Corpus: A Dutch Corpus for the Detection of Age, Gender, Personality, Sentiment and Deception in Text. In: Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014). Reykjavik, Iceland (2014)
65. Villena-Román, J., González-Cristóbal, J.C.: DAEDALUS at PAN 2014: Guessing Tweet Author's Gender and Age—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
66. Wang, H., Lu, Y., Zhai, C.: Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 783–792 (2010)
67. Weren, E.R., Moreira, V.P., de Oliveira, J.P.: Exploring Information Retrieval Features for Author Profiling—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
68. Williams, K., Chen, H.H., Giles, C.: Supervised Ranking for Plagiarism Source Retrieval—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]
69. Zhang, C., Zhang, P.: Predicting Gender from Blog Posts. Technical Report. University of Massachusetts Amherst, USA (2010)
70. Zubarev, D., Sochenkov, I.: Using Sentence Similarity Measure for Plagiarism Source Retrieval—Notebook for PAN at CLEF 2014. In: Cappellato et al. [6]