The Artificial Intelligence Workbench: a retrospective review

Hugo López-Fernández[a], Miguel Reboiro-Jato[a], José A. Pérez Rodríguez[b], Florentino Fdez-Riverola[a], Daniel Glez-Peña[a]

[a] Escuela Superior de Ingeniería Informática, University of Vigo, Edificio Politécnico, Campus Universitario As Lagoas s/n, 32004 Ourense, Spain.

[b]CFR: Centro de Formación e Recursos de Ourense, Rúa Universidade s/n, 32005 Ourense, Spain.

hlfernandez@uvigo.es, mrjato@uvigo.es, jose.a.perez@edu.xunta.es, riverola@uvigo.es, dgpena@uvigo.es

KEYWORD

Scientific software development; biomedical informatics; open software; application framework; artificial Intelligence; AIBench

ABSTRACT

Last decade, biomedical and bioinformatics researchers have been demanding advanced and user-friendly applications for real use in practice. In this context, the Artificial Intelligence Workbench, an open-source Java desktop application framework for scientific software development, emerged with the goal of providing support to both fundamental and applied research in the domain of translational biomedicine and bioinformatics. AIBench automatically provides functionalities that are common to scientific applications, such as user parameter definition, logging facilities, multi-threading execution, experiment repeatability, workflow management, and fast user interface development, among others. Moreover, AIBench promotes a reusable component based architecture, which also allows assembling new applications by the reuse of libraries from existing projects or third-party software. Ten years have passed since the first release of AIBench, so it is time to look back and check if it has fulfilled the purposes for which it was conceived to and how it evolved over time.

1. Introduction

During last decade, there has been a growing interest in the development of translational software applications that make use of artificial intelligence (AI) techniques to analyze biological data and, therefore, it has been an active research area. Two main factors lead to this boom of bioinformatics and biomedicine: (*i*) reduction in costs of experiments, and (*ii*) increase in capacity and performance of data analysis machines (i.e., high-throughput data analysis). Nowadays, vast amounts of data are available and computational applications play a key role in its analysis and processing. Researchers demand fast and user-friendly applications containing advanced and specific functionalities that allow them to make use of AI techniques to analyze data and extract biologically meaningful knowledge.

However, development of such applications in an applied research context faces a large number of particular requisites ranging from computational requirements to usability, apart from those related to software engineering. Specific aspects include: (*i*) heterogeneous data exchanging, (*ii*) integration of third-party or previously developed algorithms, (*iii*) multi-platform compatibility, (*iv*) capability of repeating workflows while changing inputs or parameter settings, (*v*) use of logging messages to monitor long operations, (*vi*) setting a high and variable number of parameters before running experiments, or (*vii*) taking advantage of multi-threading capabilities in highly demanding operations, among others.

In this context, the Artificial Intelligence Workbench (AIBench) is an open-source Java desktop application framework, specifically intended to improve both quality and productivity in the development of specialized applications for computer-assisted biomedical and clinical research [Glez-Peña et al., 2010]. AIBench was firstly released in 2006 with the main objective of covering the gap between general-purpose desktop application frameworks and the specific requirements of scientific software development. To accomplish this ambitious objective, AIBench allows rapid development of high quality application prototypes with minimum effort into problem-unrelated functionalities and with the maximum reusability level of previously coded algorithms. Our framework is particularly useful for the implementation of novel workflows - or refactoring of existing tools - characterized by the utilization of the input-process-output (IPO) model, which is representative of different well-known AI techniques [Alvarado-Pérez et al., 2015]. Some examples of previously developed applications that could benefit from AIBench range from different domains such as ambient assisted living (AAL) [Pereira et al., 2012] or biotechnology [Santos et al., 2012], to more specific tasks like program supervision [Khayati and Lejouad, 2013]. The large number of successfully scientific applications developed using AIBench [López-Fernández et al., 2011a; Fdez-Riverola et al., 2012] serves as proof of its usefulness, as we will see in detail afterwards.

In order to easily develop any application with AIBench, the developer structures the problem using three types of components: (*i*) *operations*, containing the application main logic, (*ii*) *datatypes*, used as input or output of the *operations*, and (*iii*) *views*, able to render *datatypes* in a user-friendly manner. The framework automatically provides common functionalities present in typical scientific applications, such as user parameter definition, logging facilities, multi-threading execution, experiment repeatability, *workflow management, and fast user interface development*, among others. Moreover, using AIBench encourages the implementation of reusable components, thus allowing assembling new applications by reusing libraries from past projects or third-party software. Moreover, AIBench is also suitable for adapting existing tools so that, for example, it is possible to convert a console tool or program into a graphical application [López-Fernández et al., 2011b].

Ten years have passed since the first release of AIBench, so it is time to look back and check if the developed framework has fulfilled the purposes for which it was initially conceived to and how it evolved over time. The purpose of this retrospective review paper is to present the most relevant changes in the AIBench project and to review how it has been used in the development of biomedicine and bioinformatics applications, highlighting those where artificial intelligence methods are applied.

2. Framework evolution

This section aims to reflect the evolution of our framework since its first release in 2006. Figure 1 shows a timeline containing major relevant milestones of the AIBench project, along with the most important AIBench-based developed applications. Three important aspects concerning the evolution of the framework are commented in this section: (*i*) project building management, (*ii*) version control system, and (*iii*) plugin development.

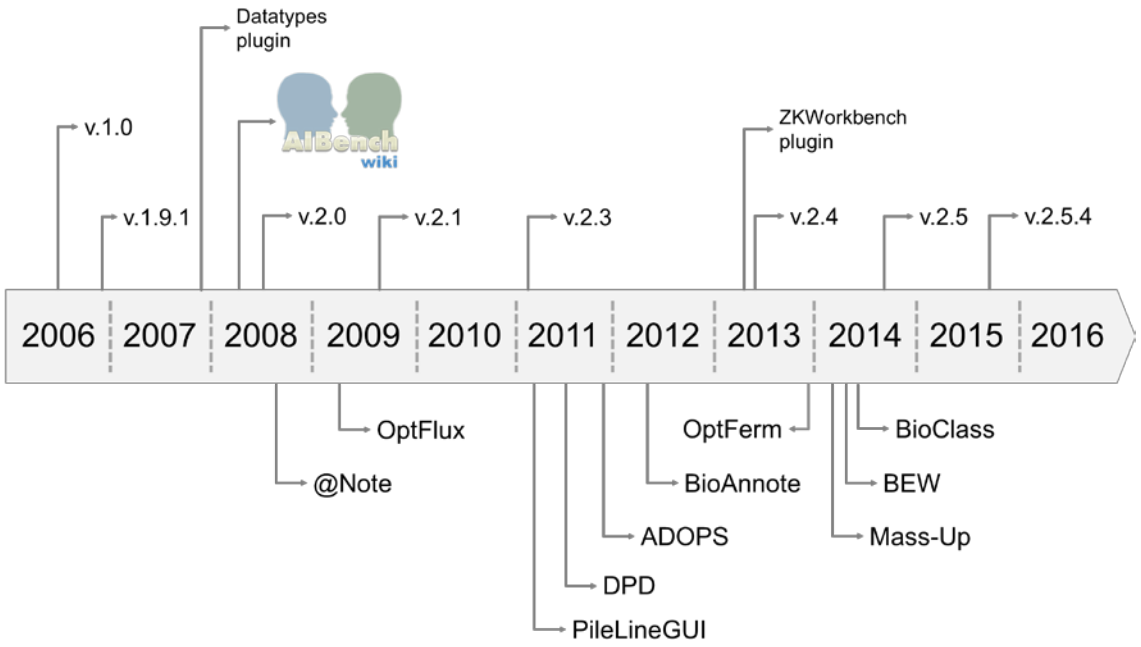

Figure 1: AIBench project timeline. First release was made by the beginning of 2006. To date, there are more than fifteen applications developed with AIBench.

2.1. Project building

In software development, it is crucial to have an automated procedure to transform source code into a final application that can be successfully deployed, installed or distributed [Spinellis, 2008]. At the beginning, AIBench was automatically built using Apache Ant, which, at that time, was the reference build automation tool for Java projects. However, due to some Ant limitations and its high complexity in long-term projects, Java community moved towards Maven. Maven covers two key aspects of software development: (*i*) defines how software is built, and (*ii*) manages its dependencies. Unlike Ant, Maven relies on conventions and performs a very efficient dependency management, avoiding the need to store local copies of each dependency version in the source code repository and, therefore, dramatically reducing its size. Due to these reasons, in 2014 AIBench was converted into a Maven project, making it a modern framework where existing Maven projects can be easily integrated. After this adaptation, AIBench's source code repository size decreased from 100MB to 13MB.

2.2. Version control system

Another important aspect of the software development process is source code control. When AIBench project started, the Concurrent Versions System (CVS)[1] was used as version control system. At that time, it was very popular since it was open-source software, released under the GNU General Public License,

[1] http://savannah.nongnu.org/projects/cvs

and there was regular development to add features and fix bugs. However, there have been no new releases since 2008 and CVS received some criticisms: (*i*) expensive branching, (*ii*) lack of support for distributed revision control, (*iii*) not atomic commits, (*iv*) poor binary files support, and (*v*) not tracking the moving or renaming of files and directories, among others. Due to these reasons, in 2014 AIBench project was moved into Git[2], a modern, distributed version control system created by Linus Torvalds. Git is very fast and scalable, which makes it efficient for handling large projects. With Git, there is no need to have central repository to commit changes: each developer can have its own repository, and they can pull/fetch from each other repositories, in a symmetric way. However, it is common for large, open-source projects to have established a central repository from which everyone pull from, that contains the stable releases. This change also propitiated the publication of AIBench's source code in Github[3] by the end of 2015.

As a summary of the work done, Figure 2 shows the number of commits by year since the start of the AIBench project. The highest activity corresponds to the first years of development, from 2006 to 2009, where the framework reached its maturity. Since 2010, there is also a significant activity corresponding to bug fixes and new features, which is a good indicator of AIBench's good health.

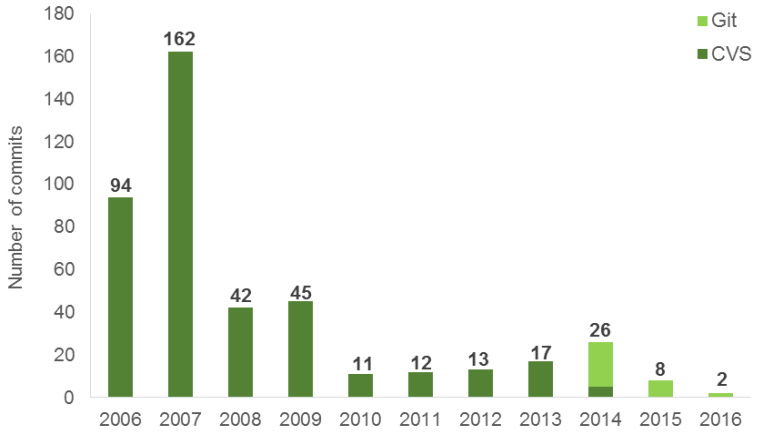

Figure 2: Number of commits by year in the AIBench project.

2.3. Plugins

One of the most important aspects of AIBench's design is its plugin-based architecture[4]. This means that both the core framework components (i.e., the Core, the Workbench, and the built-in services) and the final application functionalities are implemented as plugins. This allows applications to be modularized by packaging code, resources and a plugin descriptor into self-contained plugins, reusable for different developments. Plugins presented in this section contributed to AIBench's ecosystem, being used by several applications and giving new possibilities to them.

One of the first public plugins was the *SING Datatypes* plugin, which contains a collection of useful datatypes and views for the straightforward application of different AI techniques. The *Gaggle Tools*

[2] https://git-scm.com/
[3] https://github.com/sing-group/aibench-project
[4] http://www.aibench.org/index.php?option=com_content&view=section&id=5&Itemid=59

plugin allows the integration of an AIBench application within a *Gaggled* environment[5], so that it has the capability for exchanging data between independently developed software tools and databases and, ultimately, to enable interactive exploration of systems biology data. The *AIBench-Chart* plugin provides views to create and customize different types of charts from *Data* objects belonging the *SING Datatypes* plugin. Although there is not an official release, the *RPlugin* is 'de facto' shared among several applications. This plugin allows connecting AIBench applications to the R statistical package[6] through the rJava library.

Finally, the *ZKWorkbench* plugin gives the possibility of delivering web applications using the AIBench framework within the same philosophy of every AIBench application. This plugin also gives the possibility of converting AIBench-based desktop applications into web applications by only creating AIBench web-adapted *views* while reusing *operations* and *datatypes*.

3. AIBench-based developed tools

As it can be seen in Figure 1, many AIBench-based applications have been developed since its first public release. Although some of them do not incorporate AI techniques, they are still relevant due to their contributions to the framework's evolution and because they represent a valuable demonstration about its applicability to the development of translational bioinformatics applications. For this reason, in this section we review the most relevant ones.

3.1. Merlin

The *Metabolic Models Reconstruction Using Genome-Scale Information* (Merlin) tool is a user-friendly AIBench-based application that helps in the reconstruction of genome-scale metabolic models for any single organism that has its genome sequenced [Dias et al., 2015]. *Merlin* is a mono-plugin application that integrates more than a hundred of libraries. An interesting fact is how authors extended the initial version of *Merlin* to deal with metagenomics data [Barbosa et al., 2014]. This extension was easier due to the reusable and easy-to-extend architecture promoted by AIBench.

3.2. BEW

The *Biofilms Experiment Workbench* (BEW) is a novel software workbench for the operation and analysis of biofilms experimental data, which supports full management and analysis inter-lab experiments [Pérez-Rodríguez et al., 2015]. *BEW* integrates third-party software in order to allow users to perform routine data management, as well as flexible data analyses. Among its functionalities, it allows performing statistical tests using the R statistical package. In order to support the communication with R, this tool incorporates the *RPlugin* for AIBench, which is responsible for the communication between AIBench and R. This plugin is a powerful example of component reusability, since it is shared between several AIBench-based applications that make use of R routines.

[5] http://gaggle.systemsbiology.net/docs/
[6] https://www.r-project.org/

3.3. BioAnnote

BioAnnote implements a software platform for annotating biomedical documents by using different high-quality online resources such as Medlineplus and Freebase, with application in medical learning environments [López-Fernández et al., 2013a]. In *BioAnnote*, users can introduce biomedical documents and obtain annotations for the relevant terms, such as diseases, symptoms and treatments. In addition, users can retrieve more detailed information for each recognized entity (making use of an integrated web browser) including related topics, external links, related bibliography and paper abstracts. This application is a good candidate to be extended using the *ZKWorkbench* so that all the BioAnnote functionalities can also be served as a web application. To do that, the annotation engine (i.e., *operations*) and *datatypes* would be completely reused, and it only would be necessary to implement the web-adapted versions of existing *views*. Finally, one of the more useful features of *BioAnnote* is the possibility to annotate several documents in a row using the batch mode. To implement this feature, *BioAnnote* takes advantage of the AIBench built-in *BeanShell* plugin.

3.4. ADOPS

Automatic Detection of Positively Selected Sites (ADOPS) represents a novel software pipeline implemented with the goal of providing an automatic and flexible tool for detecting positively selected amino acid sites from a set of unaligned nucleotide sequence data [Reboiro-Jato et al., 2012]. This application is interesting because it implements a complex pipeline by integrating third-party software, being the first genomics pipeline developed with AIBench.

3.5. PileLineGUI

PileLineGUI [López-Fernández et al., 2011b] offers an intuitive graphical desktop application that servers as front-end of *PileLine* [Glez-Peña et al., 2011], a flexible command-line toolkit for efficient handling, filtering, and comparison of genomic position files produced by next-generation sequencing experiments. The *PileLine* toolbox lacks of a graphical user interface (GUI) allowing biologists without advanced informatics skills to use it, so that the AIBench framework was used to create a front-end that conceals the command-line complexity. In this example, the AIBench IPO model allowed to straightforwardly create a GUI (Figure 3) so that: (*i*) each command was wrapped by an *operation*, (*ii*) each input/output data format was wrapped by a *datatype*, and (*iii*) there is a generic tabular *view* to explore results. Moreover, *PileLineGUI* includes a new Java-based genome browser (i.e., a *view*) as a separate plugin to facilitate its later inclusion and reuse in other projects.

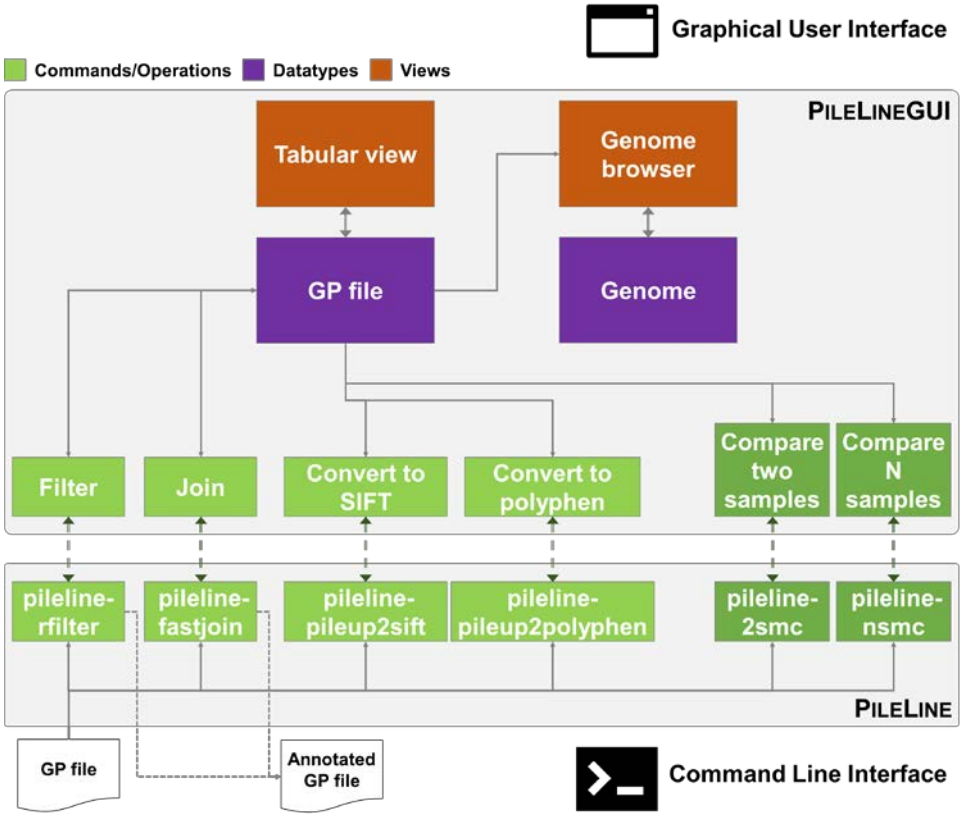

Figure 3: Integration between PileLine and PileLineGUI. Note how PileLine commands (green) are directly wrapped by AIBench's operations. This way, a GUI is easily developed for an existing command line application.

3.6. Decision Peptide Driven

The *Decision Peptide-Driven* (DPD) tool implements an application for assisting researchers in a specific protocol for protein quantification based on 18O inverse labeling and Matrix Assisted Laser Desorption/Ionization (MALDI-TOF) mass spectrometry (MS) analysis [Santos et al., 2010]. This protocol compares the results of the direct and inverse experiments and quickly identifies reproducible peptides (i.e., those that have similar direct and inverse ratios) that could be used to subsequent and accurate protein absolute quantification. This development is noteworthy because it established a starting point for future developments of MALDI-TOF MS data analysis applications.

3.7. MLibrary

The *MLibrary* software tool provides a database search engine developed to assist the user in the detection and identification of small compounds, such as anabolic androgenic steroids (AAS) using MALDI-TOF MS [Galesio et al., 2013]. *MLibrary* compares the results of the experimental data against those theoretic values included in the database. It simplifies the interpretation of MS data and reduces the

amount of time required to manually analyze the results. Regarding AIBench, this tool contributes, together with *DPD*, to the establishing a starting point for more advanced MALDI-TOF data analysis applications.

4. Application of AI techniques

This section aims to review those AIBench-based applications that include AI algorithms and together with different studies where AIBench was used to support the application of AI techniques to solve biological problems.

4.1. Microarray classification

As stated previously, one advantage of using AIBench is the rapid development of functional applications by allowing developers to concentrate in problem-related components. A good example of this feature is demonstrated in two studies in microarray classification, where authors used AIBench to develop a platform that supports novel classification techniques [Reboiro-Jato et al., 2014; Reboiro-Jato et al., 2013]. As this kind of problems fits the Input (microarray data) – Processing (application of classification techniques) – Output (classification performance metrics) model, the usage of AIBench was straightforward and allowed to save a lot of time, by concentrating in the development of advanced classification techniques rather than unrelated features (e.g., parameter definition, logging facilities or experiment repeatability). These applications were also important because a plugin for the Weka software [Hall et al., 2009] that facilitates the use of this well-established machine learning environment was developed. As we will see below, this plugin was also reused in other complementary studies and applications following AIBench's philosophy.

4.2. MALDI-TOF mass spectrometry data analysis

As commented in the previous section, AIBench has been successfully used for the development of proteomics applications such as *MLibrary* or *DPD*. These applications did not make use of AI to analyze data, but they designed and put together related components (e.g., peak list loading, basic spectrum viewer) that were reused in subsequent studies and tools applying AI and ML techniques to the analysis of MALDI-TOF MS data.

The first two studies focus on classification tasks. By one hand, the adequacy of MALDI-TOF MS for wine classification was evaluated [Nunes-Miranda et al., 2012]. In this study, AIBench was used to execute all runs of an ad hoc LOO (Leave-One-Out) cross-validation experiment using the Weka plugin. On the other hand, the influence of low-level MALDI-TOF data preprocessing techniques on sample classification was analyzed [López-Fernández et al., 2014]. Since preprocessing is applied to MALDI-TOF raw data, this study introduced a new AIBench plugin for its preprocessing (using the *RPlugin*) and expanded existing plugins to load and visualize raw data from different formats. As we can realize, these studies reused and improved previously developed plugins.

Taking advantage of existing MS plugins, the *BiMS* software package implemented a novel workflow for the application of biclustering to mass spectrometry data [López-Fernández et al., 2013b]. This application incorporated a new plugin to support the use of biclustering, a data mining technique that allows simultaneous clustering of the rows and columns of a matrix, to analyzing MALDI-TOF MS data.

Finally, *Mass-Up*[7], an open-source mass spectrometry utility for proteomics designed to support the preprocessing and AI-based analysis of MALDI-TOF mass spectrometry data [López-Fernández et al., 2016], was developed. By taking advantage of previously existing plugins, *Mass-Up* was specifically created to perform complete analyses of MALDI-TOF MS data [López-Fernández et al., 2015], allowing the users to: (*i*) import raw data from different formats (mzML, mzXML, csv); (*ii*) carry out its full preprocessing; (*iii*) perform statistical analysis and tests, and (*iv*) perform different types of AI-based analyses, including supervised (e.g., biomarker discovery, predictor building, etc.) and unsupervised (e.g., clustering, biclustering, etc.) techniques.

4.3. Optimization of fed-batch fermentation processes

OptFerm[8] stands for an open-source and user-friendly computational platform for the simulation and optimization of fermentation processes [Rocha et al., 2015]. This tool is focused on optimizing a feeding trajectory to be tested into a fed-batch bioreactor and to calculate the best concentration of nutrients to initiate the fermentation. To do that, authors introduced in *OptFerm* different bio-inspired artificial intelligence algorithms, such as Evolutionary Algorithms (EAs), Differential Evolution (DE) and Particle Swarm Optimization (PSO).

4.4. In silico metabolic engineering

OptFlux[9] is another open-source and modular software to support in silico metabolic engineering tasks aimed at being the reference computational application in the field. *OptFlux* was released as a novel software tool that implements methods for the phenotype simulation and optimization of microbial strains using integrated models, encompassing both metabolic and regulatory information [Rocha et al., 2010]. Regarding the strain optimization tasks, this tool is able to run two AI-based optimization methods: Simulating Annealing (SA), and Evolutionary Algorithms [Vilaça et al., 2011]. Thanks to its AIBench-based plugin architecture, it was easily extended to include new AI-based methods that allow the identification of sets of genes being over/under expressed [Gonçalves et al., 2012], as well as a plugin for the visualization of metabolic models [Noronha et al., 2013].

4.5. Biomedical text mining

Previously, we mentioned *BioAnnote*, a tool for annotating biomedical texts by using different high-quality online resources such as Medlineplus and Freebase. In this section, we cover two complementary AIBench-based applications that also deal with biomedical texts, but allowing more advanced analyses based in text mining and text classification techniques: *@Note* and *BioClass*.

@Note[10] is a biomedical text mining platform that copes with major information retrieval and extraction tasks and promotes multi-disciplinary research [Lourenço et al., 2009]. The use of AIBench allowed the authors to accomplish their design principles for *@Note*: interoperability, flexibility and modularity. The first release of *@Note* included three main plugins: *@Note* core plugin, GATE plugin,

[7] http://sing.ei.uvigo.es/mass-up/
[8] http://darwin.di.uminho.pt/optferm/
[9] http://www.optflux.org/
[10] http://sysbio.di.uminho.pt/anote/wiki

and YALE plugin. While core plugin integrates author developments, the other two plugins adapt well-known open-source text mining and ML algorithms. The latest version of *@Note* (2.1.1, 16 February 2016) introduced the *BioTML* module, a ML-based plugin that allows users to create models, train them based on available annotations from the available manual curation environment, and further use them in a prediction stage to annotate documents from a selected *@Note* corpus.

BioClass[11] is another AIBench-based tool for biomedical text classification [Romero et al., 2014]. Including different AI and ML algorithms, it allows researchers to split a document set (directly related with a specific topic) into relevant or irrelevant documents. Again, thanks to the plugin architecture of this tool, authors were able to effortlessly introduce in *BioClass* a novel method for the classification of biomedical text documents based on Hidden Markov Models [Seara Vieira et al., 2014]. Additionally, as in the microarray classification case, this platform served authors to evaluate and compare different classification techniques.

5. Conclusions

In this contribution, we have reviewed the Artificial Intelligence Workbench, originally designed to support the rapid development of fully functional translational applications in bioinformatics and biomedical research. In view of the results, AIBench has largely met this purpose: it served as base for the development of a vast number of applications and supported several studies where different AI methods were successfully applied.

As we have seen, during the last decade AIBench has adequately evolved and maintained to date, which is a healthy sign. The project has being converted from Apache Ant and CVS into Maven and Git and, more interestingly, it has been provided with a plugin to automatically deliver web-based applications.

A core strength of AIBench is that it allows developers to easily integrate existing libraries and third-party components into a GUI-based application. In this context, our recommendations to develop novel AIBench-based applications are: (*i*) put the application logic and core algorithms in a separate project along with unit tests, (*ii*) locate GUI components in a separate project that depends in the *core* project, and (*iii*) create an AIBench project that integrates *core* and *gui* projects. This way, *core* and *gui* projects are totally AIBench-independent and can be easily reused in other developments. Additionally, the AIBench project will contain only the minimum framework-related code needed to create the application.

Acknowledgements

The authors would like to thank all the valuable comments, ideas and discussion from the high number of AIBench developers who have contributed towards the improvement of our framework since its inception. H. López-Fernández is supported by a postdoctoral fellowship from Xunta de Galicia. SING group thanks CITI (*Centro de Investigación, Transferencia e Innovación*) from University of Vigo for hosting its IT infrastructure.

[11] http://sing.ei.uvigo.es/bioclass/

6. References

Alvarado-Pérez, J.C., Peluffo-Ordó-ez, D.H., and Theron, R., 2015, Bridging the Gap between Human Knowledge and Machine Learning. Advances in Distributed Computing and Artificial Intelli-gence Journal, 4(1):54-64. http://dx.doi.org/10.14201/ADCAIJ2015415464

Barbosa, P., Dias, O., Arrais, J.P., and Rocha, M., 2014, Metagenomic Analysis of the Saliva Microbiome with Merlin. Advances in Intelligent Systems and Computing, 294:191-199. http://dx.doi.org/10.1007/978-3-319-07581-5_23

Dias, D., Rocha, M., Ferreira, E.C., and Rocha, I., 2015, Reconstructing genome-scale metabolic models with merlin. Nucleic Acids Research, 43(8):3899-3910. http://dx.doi.org/10.1093/nar/gkv294

Fdez-Riverola, F., Glez-Pe-a, D., López-Fernández, H., Reboiro-Jato, M., and Méndez J.R., 2012, A Java application framework for scientific software development. Software: Practice & Experience, 42/8:1015-1036. http://dx.doi.org/10.1002/spe.1108

Galesio, M., López-Fdez, H., Reboiro-Jato, M., Gómez-Meire, S., Glez-Pe-a, D., Fdez-Riverola, F., Lodei-ro, C., Diniz, D., and Capelo, J.L., 2013, Speeding up the screening of steroids in urine: develop-ment of a user-friendly library. Steroids, 78(12-13):1226-1232. http://dx.doi.org/10.1016/j.steroids.2013.08.014

Glez-Pe-a, G., Reboiro-Jato, M., Maia, P., Díaz, F., and Fdez-Riverola, F., 2010, AIBench: a rapid appli-cation development framework for translational research in biomedicine. Computer Methods and Programs in Biomedicine, 98(2010):191-203. http://dx.doi.org/10.1016/j.cmpb.2009.12.003

Glez-Pe-a, D., Gómez-López, G., Reboiro-Jato, M., Fdez-Riverola, F., and Pisano, D.G., 2011, PileLine: a toolbox to handle genome position information in next-generation sequencing studies. BMC Bio-informatics, 12:31. http://dx.doi.org/10.1186/1471-2105-12-31

Gonçalves, E., Rocha, I., and Rocha, M., 2012, Computational tools for strain optimization by tuning the optimal level of gene expression. Advances in Intelligent and Soft Computing, 154:251-258. http://dx.doi.org/10.1007/978-3-642-28839-5_29

Hall, M., Frank, E., Geoffrey, H., Pfahringer, B., Reutemann, P., and Witten, I.H., 2009, The WEKA Data Mining Software: An Update. SIGKDD Explorations, 11:10–18. http://dx.doi.org/10.1145/1656274.1656278

Khayati, N., and Lejouad-Chaarib, W., 2013. A Distributed and Collaborative Intelligent System for Medical Diagnosis. Advances in Distributed Computing and Artificial Intelligence Journal, 2(2):1-16.

López-Fernández, H., Reboiro-Jato, M., Glez-Pe-a, D., Méndez-Reboredo, J.R., Santos, H.M., Carreira, R.J., Capelo, J.L., and Fdez-Riverola, F., 2011a, Rapid development of proteomic applications with the AIBench framework. Journal of Integrative Bioinformatics, 8/3:171.

López-Fernández, H., Glez-Pe-a, D., Reboiro-Jato, M., Gómez-López, G., Pisano, D.G., and Fdez-Riverola, F., 2011b, PileLineGUI: a desktop environment for handling genome position files in next-generation sequencing studies. Nucleic Acids Research, 39(suppl. 2):W562-W566. http://dx.doi.org/10.1093/nar/gkr439

López-Fernández, H., Glez-Pe-a, D., Reboiro-Jato, M., Aparicio, F., Gachet, D., Buenaga, M., and Fdez-Riverola, F., 2013a, BioAnnote: A software platform for annotating biomedical documents with application in medical learning environments. Computer Methods and Programs in Biomedicine, 111/1:139-147. http://dx.doi.org/10.1016/j.cmpb.2013.03.007

López-Fernández, H., Reboiro-Jato, M., Madeira, S.C., López-Cortés, R., Nunes-Miranda, J.D., Santos, H.M., Fdez-Riverola, F., and Glez-Pe-a, D., 2013b, A workflow for the application of biclustering to mass spectrometry data. Advances in Intelligent Systems and Computing, 222: 145-153. http://dx.doi.org/10.1007/978-3-319-00578-2_19

López-Fernández, H., Reboiro-Jato, M., Glez-Pe-a, D., and Fdez-Riverola, F., 2014, A comprehensive analysis about the influence of low-level preprocessing techniques on mass spectrometry data for sample classification. International Journal of Data Mining and Bioinformatics, 10/4:455-473.
http://dx.doi.org/10.1504/IJDMB.2014.064897

López-Fernández, H., Santos, H.M., Capelo, J.L., Fdez-Riverola, F., Glez-Pe-a, D., and Reboiro-Jato, M., 2015, Mass-Up: an all-in-one open software application for MALDI-TOF mass spectrometry knowledge discovery. BMC Bioinformatics, 16:318.
http://dx.doi.org/10.1186/s12859-015-0752-4

López-Fernández, H., 2016, Application of data mining and artificial intelligence techniques to mass spectrometry data for knowledge discovery. Revista Iberoamericana de Inteligencia Artificial, 19(57):22-25.

Lourenço, A., Carreira, R., Carneiro, S., Maia, P., Glez-Pe-a, D., Fdez-Riverola, F., Ferreira, E.C., Rocha, I., and Rocha, M., 2009, @Note: A workbench for Biomedical Text Mining. Journal of Biomedi-cal Informatics, 42(4): 710-720.
http://dx.doi.org/10.1016/j.jbi.2009.04.002

Noronha, A., Vilaça, P., and Rocha, M., 2013, Network Visualization Tools to Enhance Metabolic Engi-neering Platforms. Advances in Intelligent Systems and Computing, 222:137-144.
http://dx.doi.org/10.1007/978-3-319-00578-2_18

Nunes-Miranda, J.D., Santos, H.M., Reboiro-Jato, M., Fdez-Riverola, F., Igrejas, G., Lodeiro C., and Capelo, J.L., 2012, Direct matrix assisted laser desorption ionization mass spectrometry-based analysis of wine as a powerful tool for classification purposes. Talanta, 91:72-76.
http://dx.doi.org/10.1016/j.talanta.2012.01.017

Pereira, A., Felisberto, F., Maduro, L., and Felgueiras, M., 2012. Fall Detection on Ambient Assisted Living using a Wireless Sensor Network. Advances in Distributed Computing and Artificial Intelligence Journal, 1(1):62-77.

Pérez-Rodríguez, G., Glez-Pe-a, D., Azevedo, N.F., Pereira, M.O., Fdez-Riverola, F., and Lourenço, A., 2015, Enabling systematic, harmonised and large-scale biofilms data computation: the Biofilms Experiment Workbench. Computer Methods and Programs in Biomedicine, 118/3:309-321.
http://dx.doi.org/10.1016/j.cmpb.2014.12.005

Reboiro-Jato, D., Reboiro-Jato, M., Fdez-Riverola, F., Vieira, C.P., Fonseca, N.A., and Vieira, J., 2012, ADOPS - Automatic detection of positively selected sites. Journal of Integrative Bioinformatics, 9/3:200.

Reboiro-Jato, M., Laza, R., López-Fernández, H., Glez-Pe-a, D., Díaz, F., and Fdez-Riverola, F., 2013, genEnsemble: a new model for the combination of classifiers and integration of biological knowledge applied to genomic data. Expert Systems With Applications, 40/1:52-63.
http://dx.doi.org/10.1016/j.eswa.2012.07.003

Reboiro-Jato, M., Díaz, F., Glez-Pe-a, D., and Fdez-Riverola, F., 2014, A novel ensemble of classifiers that use biological relevant gene sets for microarray classification. Applied Soft Computing, 17:117-126.
http://dx.doi.org/10.1016/j.asoc.2014.01.002

Rocha, I., Maia, P., Evangelista, P., Vilaça, P., Soares, S., Pinto, J.P., Nielsen, J., Patil, K.R., Ferreira, E.C., and Rocha, M., 2010, OptFlux: An open-source software platform for in silico metabolic engineer-ing. BMC Systems Biology, 4:45.
http://dx.doi.org/10.1186/1752-0509-4-45

Rocha, M., Mendes, R., Rocha, O., Rocha, I., and Ferreira, E.C., 2014, Optimization of fed-batch fer-mentation processes with bio-inspired algorithms. Expert Systems With Applications, 41(5): 2186–2195.
http://dx.doi.org/10.1016/j.eswa.2013.09.017

Romero, R., Seara Vieira, A., Iglesias, E. L., Borrajo, L., 2014, BioClass: A Tool for Biomedical Text Classification. Advances in Intelligent Systems and Computing, 294:243-251.
http://dx.doi.org/10.1007/978-3-319-07581-5_29

Santos, A., Nogueira, R., and Lourenço, A., 2012. Applying a text mining framework to the extraction of numerical pa-rameters from scientific literature in the biotechnology domain. Advances in Dis-tributed Computing and Artificial Intelligence Journal, 1(1):1-8.
http://dx.doi.org/10.1155/2012/312132

Santos, H.M., Reboiro-Jato, M., Glez-Pe-a, D., Diniz, M.S., Fdez-Riverola, F., Carvalho R., Lodeiro, C., and Capelo, J.L., 2010, Decision peptide-driven: a free software tool for accurate protein quantifi-cation using gel electrophoresis and matrix assisted laser desorption ionization time of flight mass spectrometry. Talanta, 82/4:1412-1420. http://dx.doi.org/10.1016/j.talanta.2010.07.007

Seara Vieira, A., Iglesias, E.L., Borrajo, L., and Romero R., 2014. A HMM Text Classification Model with Learning Capacity. Advances in Distributed Computing and Artificial Intelligence Journal, 3(3):21-34.

Spinellis, D., 2008, Software Builders. IEEE Software, 25(3):22-23. http://dx.doi.org/10.1109/MS.2008.74

Vilaça, P., Rocha, I., and Rocha, M., 2011, A computational tool for the simulation and optimization of microbial strains accounting integrated metabolic/regulatory information. Biosystems, 103(3):435-441. http://dx.doi.org/10.1016/j.biosystems.2010.11.01