Smarter interoperability based on automatic schema matching and intelligence amplification

Jean Paul Sebastian Piest[a], Lucas Onno Meertens[a,b], Johan Buis, Maria Eugenia Iacob[a] and Marten van Sinderen[a]

[a] *University of Twente, Drienerlolaan 5, Enschede, 7522 NB, The Netherlands*
[b] *CAPE Groep, Transportcentrum 16, Enschede, 7547 RW, The Netherlands*

Abstract

The process of connecting enterprise systems and applications to digital platforms and ecosystems requires interoperability on different levels and eventually leads to the task of creating a schema mapping. At present, this task is carried out manually and prior research has proven this task difficult to automate. In this paper, we discuss the suitability of machine learning approaches to create an auto-mapping functionality, so different schemas and standards can (partially) be mapped automatically, and incorporate a symbiotic approach to improve the matching result. To the best of our knowledge, this is a new approach with potential to reduce the time needed for schema matching tasks. The main contribution of this paper is a reference architecture and prototype for smarter interoperability using a combination of automatic schema matching, based on machine learning, and intelligence amplification.

Keywords 1

Interoperability, machine learning, schema matching, auto-mapping, intelligence amplification

1. Introduction

Research into schema matching mostly focusses on systems that can propose possible matches without human involvement [3]. At present this task is carried out manually, which is time consuming and error prone [3][12]. Moreover, automation is difficult because the exact semantics of the data are only completely understood by the designers of the schema, and not fully captured by the schema itself [18]. Many approaches cannot guarantee the correctness of the mappings, which could have a cascade effect and lead to further errors [16]. When accuracy is important, user intervention during the matching process becomes essential, and earlier research indicates that this intervention could significantly improve the matching result [16]. Intelligence Amplification (IA) has been gaining more prominence, but validated research in this area is still in its infancy [13]. In this paper, we present a reference architecture IA driven schema matching. This research is conducted based on the Design Science Research Methodology for Information Systems Research [2].

2. Related work

Schema matching is a basic problem in many data-processing applications, such as data integration, data warehousing, and semantic query processing [3]. It aims at identifying semantic correspondences between metadata structures or models, such as database schemas or XML message definitions [20]. More precise, schema matching is the problem of pairing attributes from a source data model (from here on called schema) with attributes of a target schema such that paired attributes are likely to be semantically related [5]. Many approaches exist for schema matching and a taxonomy is provided by

Proceedings of the Workshops of I-ESA 2020, 17-11-2020, Tarbes, France
EMAIL: j.p.s.piest@utwente.nl (J.P.S. Piest); l.o.meertens@utwente.nl (L.O. Meertens); johanbuis@outlook.com (J. Buis); m.e.iacob@utwente.nl (M.E. Iacob); m.j.vansinderen@utwente.nl (M. van Sinderen);
ORCID: 0000-0002-0995-6813 (J.P.S. Piest); 0000-0002-4004-0117 (M.E. Iacob) 0000-0001-7118-1353 (M. van Sinderen);


[3]. A summary of methods developed prior to 2001 can be found in [3] and a general workflow of a schema matcher is described in [20]. A well-known example of a schema matcher is COMA [11]. COMA uses heuristics to combine the result of different matching algorithms to determine matching instances. A more recent advancement using machine learning is Yet Another Matcher (YAM) [12]. YAM is a schema matcher generator designed to create a tailor-made matcher when making a new mapping. Internally, YAM stores a repository of schemas (training data), classifiers, and similarity measures. When new schemas are presented, these are used to generate new matchers. Additional user preferences can be included by the user.

The early days of automation started with the objective of examining which tasks could best be replaced by computers because the computer could do it better cheaper [6]. One of the research fields seeking to achieve this objective is the field of Artificial Intelligence (AI) [14]. The focus of AI is to replace human reasoning, which is not always possible [9]. IA takes a different view, in which the core idea is not to replace human tasks by computer tasks, but to see how tasks can best be allocated to humans, to computers, or to both of them [6] in order to enhance human decision-making and problem-solving capabilities [1]. At present, there is no universally accepted definition for IA [1] [8] [9] [17]. We consider human empowerment as the raison d'être of IA, decision-making and problem-solving as main goals, where IA uses AI and symbiosis as means to achieve these goals. IA systems are computational systems performing some sort of intelligent decision making, based on cooperation provided by an ongoing dialogue between a human user and a computer system [10]. [7] investigated how information can effectively be shared between humans and machines. For this purpose, they propose the Proactive Autonomy Collaboration Toolkit (PACT) model. The PACT model has four primary elements: goals, work product, context, and information.

3. Reference architecture

Using the literature, we designed a reference architecture for IA-driven schema matching using the ArchiMate language [4] as shown in Figure 1. The business layer consists of three main functions of schema matching: create a mapping profile, refine (intermediate) mapping candidates, and generate a schema matcher. These functions are supported by application services of the IA system components: an Automapping Application, a Machine Learning Application, and a Software Agent. Following the PACT model, the process of creating a mapping profile starts with the user indicating the initial goals and preferences. Next, the user inputs schemas and data instances in the Automapping Application, assuming the ecosystem to which the user's organization will connect has a known data schema. The application first performs a lookup in a repository and tries to clean the inputs. The user is involved, in deciding which option is best, or, when no option is presented, come up with an option by himself. The process steps for pre-processing can occur multiple times. Based on the pre-processing, a profile is created with a (partial) search space and similarity measures. The user is involved during pre-processing and profiling using IA and the initial goal is realized in a collaborative environment. After the data has been pre-processed and the profile is created, the Software Agent is invoked. When invoking the Software Agent, the (partial) search space is sent using web services of the Automapping Application, and of the Software Agent.

Initially, the Machine Learning Application has the task to create, train and evaluate classifiers. When using machine learning, each pair of schema elements is considered to be a machine learning object, having as attributes the similarity values computed for a set of similarity measures attributed to these elements [12]. Compared to traditional machine learning, where user intervention is required afterwards, in this case active learning is required while the method is running [11]. The actions for generating candidate mappings can be repeated and are therefore iterative [13]. Following the PACT model, the initial, intermediate, and final (partial) search spaces, and candidate mapping are stored, and all actions and decisions are logged, so the machine learning model can be (re-)trained and learn from iterations. The Software Agent uses the web services provided by the Machine Learning Application to generate a list of candidate mappings which are presented to the user, who inspects, adjusts, and tests the results with the aim of refining the (intermediate) mapping candidates. The user can also opt for invoking the Software Agent, and let it select for which the user needs refinement. In this case, the

process steps are repeated. This loop, involving the re-invocation the Software Agent, is what distinguishes the IA approach from other existing approaches [13].

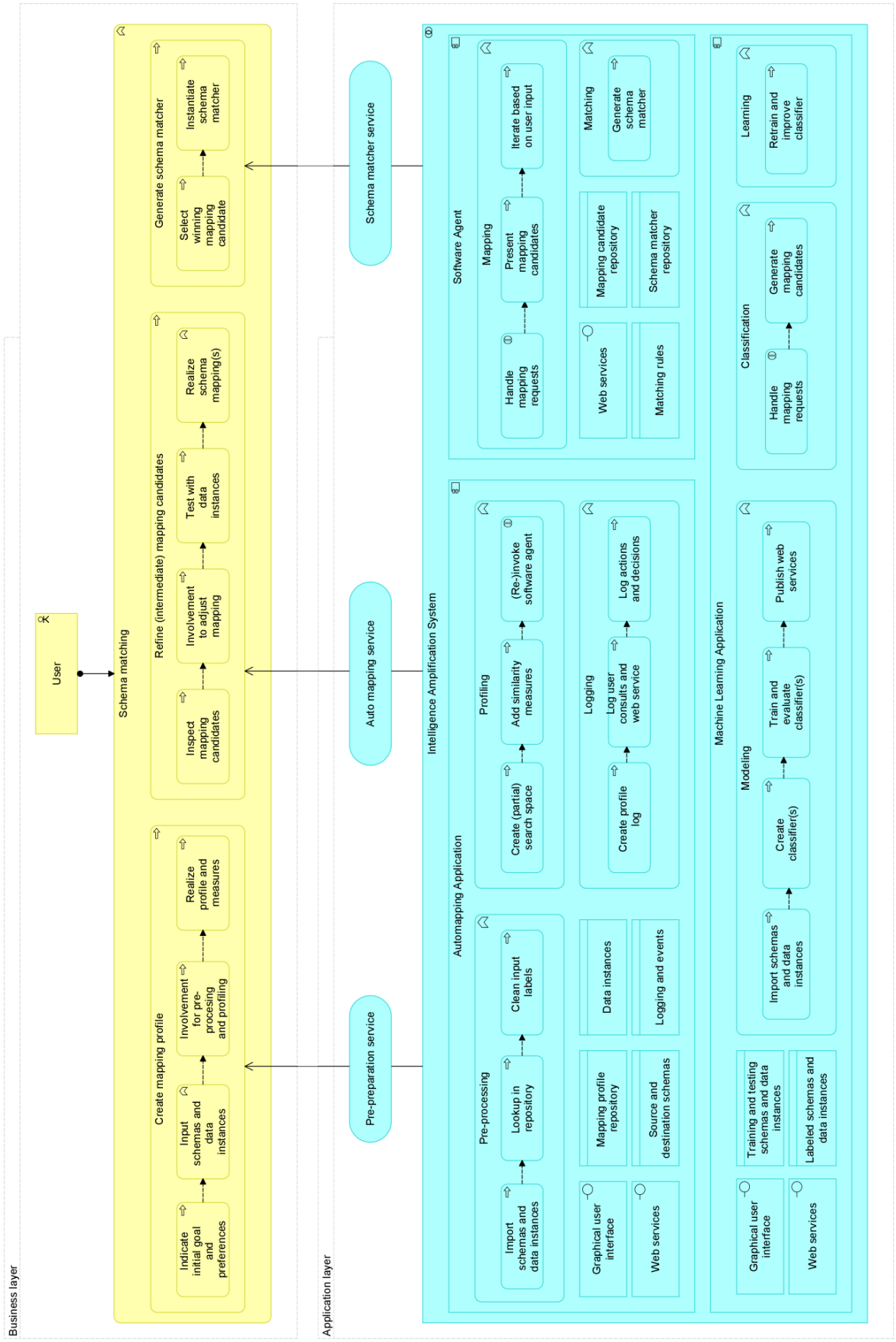

Figure 1: Reference architecture for IA driven schema matching

Also note that visualization plays an important role at this point. Presenting all schema matching correspondences to a user at once could be overwhelming and difficult to comprehend by the user, as they become frustrated scrolling through all the false positives [13]. Completing the mapping is a task that is both time consuming and cognitively demanding [13]. Explaining the reason why a specific mapping suggested by the Software agent is an important feature supporting the user, but where many approaches fall short still [13] [15]. Eventually, the winning mapping candidate is used to generate a schema matcher.

4. Prototype

Using the reference architecture, we developed a prototype of the Automapping Application. It was built using eMagiz [21] and uses the Machine Learning library provided by Microsoft [22] due to its' ease of use and implementation. We first focused on a solution that can automatically suggest mappings, and visually present the results to the user as shown in Figure 2. The involvement of the user only occurred at the end of the process, with the aim of refining the results. We tested the classifiers which are available and found that the best results can be achieved by using a neural network We created a training set by extracting existing mapping results from the mapping repository in eMagiz. Since these mappings are used in operational integrations, we consider them to be a reliable, expert verified, set. We trained the neural network to predict candidate mappings and labeled each pair instance with (1) if it has been mapped, and with (0), otherwise. This reduces the training set of 2.354.498 records to 19.486 records labelled with (1). When a user invokes the auto mapper, the same steps are followed, and the classifier is used to predict the label and the user is then presented with new mappings.

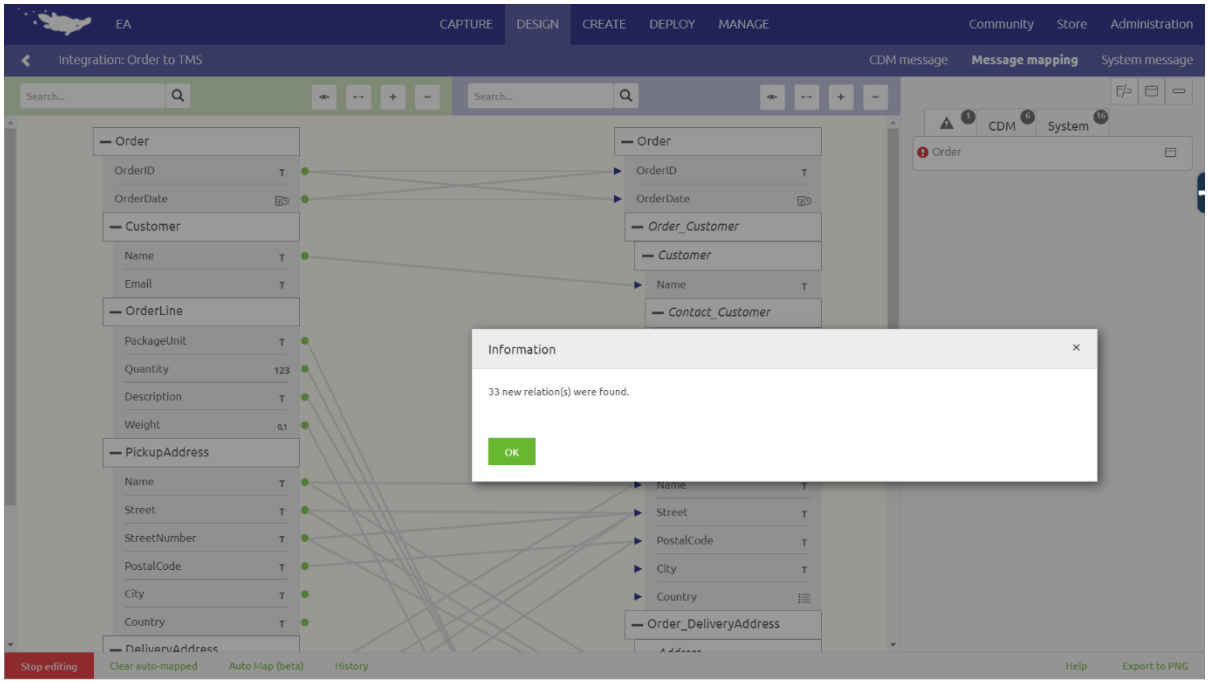

Figure 2: Prototype of the Automapping Application in eMagiz

5. Discussion

Our reference architecture provides a starting point, and a set of guidelines for developing smarter interoperability applications based on schema matching automation, the concepts of IA, and machine learning. We tried to incorporate the essence of existing architectures, but they are scarce and do not provide a complete solution for the problem we address. Therefore, the reference architecture is kept at

high abstraction since exact steps that need to be taken vary for each schema matching scenario. Preliminary results of the first design and development cycle are encouraging but require thorough validation. In the pre-processing stage, further research is needed to investigate which pre-processing steps work well and how to efficiently employ the user in the process. We believe that more attention in the pre-processing stage will be beneficial to this end. This has also been suggested in the literature. Combining multiple approaches in one matching solution, also referred to as hybrid matchers, could also lead to further improvements.

The developed prototype demonstrated that it is possible to develop an IA-driven approach to schema matching. The developed prototype however has several limitations and needs further development and testing. At this moment, entities are not taken into account. Mapping entities not only aids the completeness but may also improve results. Furthermore, we only lower the upper cases, but performance could be aided further, for example by using thesauri. Presenting the results and guiding the user with visualizations needs future work. The use of an active classifier, which can ask the user more information at runtime, is expected to be beneficial to explore improvements. Currently, we do not use the scored probability values generated. In a future version, candidate mappings can be presented to the user by giving them an option to adjust the threshold for various metrics. This allows the user to adjust the thresholds and see the impact of these changes immediately. The user is then able to select the best criteria for a given scenario. When a user is satisfied with a set of candidate mappings, he accepts them and goes to the stage of refinement.

Future research, prototyping and software development will be incorporated in the Platform Architecture for Industry 4.0 Driven Intelligence Amplification in Logistics [19].

6. Acknowledgements

This research is financially supported by the Dutch Ministry of Economic Affairs and co-financed via TKI DINALOG and NWO. Funding for this work has been granted by the ICCOS project (grant no. 2018-2-169TKI) and the CLICKS project (grant no. 439.19.633). The authors thank the anonymous reviewers for their constructive feedback.

7. References

[1] Breemen, A. J. J. van, Farkas, J. I., & Sarbo, J. J. (2011). Knowledge Representation as a Tool for Intelligence Augmentation. In Computational Modeling and Simulation of Intellect (pp. 321–341). IGI Global. https://doi.org/10.4018/978-1-60960-551-3.ch013

[2] Peffers, K., Tuunanen, T., Rothenberger, M., & Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. J. Manage. Inf. Syst., 24(January), 45–77. https://doi.org/10.2753/MIS0742-1222240302

[3] Rahm, E., & Bernstein, P. a. (2001). A survey of approaches to automatic schema matching. The VLDB Journal, 10(4), 334–350. https://doi.org/10.1007/s007780100057

[4] Iacob, M.-E., Jonkers, H., Quartel, H., Franken, H., & Berg, H. van de. (2012). Delivering enterprise architecture with TOGAF and ArchiMate. Enschede:BIZZdesign. 300 p.

[5] Assoudi, H., & Lounis, H. (2015). Coping with Uncertainty in Schema Matching: Bayesian Networks and Agent-Based Modeling Approach. In M. Benyoucef, M. Weiss, & H. Mili (Eds.) (Vol. 209, pp. 53–67). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-17957-5_4

[6] Casini, E., Depree, J., Suri, N., Bradshaw, J. M., & Nieten, T. (2015). Enhancing decision-making by leveraging human intervention in large-scale sensor networks. In 2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision (pp. 200–205). IEEE. https://doi.org/10.1109/COGSIMA.2015.7108198

[7] DiBona, P., Shilliday, A., & Barry, K. (2016). Proactive human-computer collaboration for information discovery. In B. D. Broome, T. P. Hanratty, D. L. Hall, & J. Llinas (Eds.), SPIE 9851, Next-Generation. https://doi.org/10.1117/12.2222805

[8] Dobrkovic, A., Liu, L., Iacob, M.-E., & van Hillegersberg, J. (2016). Intelligence Amplification Framework for Enhancing Scheduling Processes (pp. 89–100). https://doi.org/10.1007/978-3-319-47955-2_8
[9] Garcia, A. C. B. (2010). AGUIA: Agents Guidance for Intelligence Amplification in Goal Oriented Tasks. In 2010 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (pp. 338–344). IEEE. https://doi.org/10.1109/3PGCIC.2010.56
[10] Paraense, A. L. O., Gudwin, R. R., & Goncalves, R. de A. (2007). Designing Intelligence Augmentation System with a Semiotic-Oriented Software Development Process. In Seventh International Conference on ISDA 2007 (pp. 84–89). IEEE. https://doi.org/10.1109/ISDA.2007.16
[11] Rodrigues, D., da Silva, A., Rodrigues, R., & dos Santos, E. (2015). Using active learning techniques for improving database schema matching methods. In 2015 IJCNN (pp. 1–8). IEEE. https://doi.org/10.1109/IJCNN.2015.7280630
[12] Duchateau, F., & Bellahsene, Z. (2016). YAM: A Step Forward for Generating a Dedicated Schema Matcher. In A. Hameurlain, J. Küng, & R. Wagner (Eds.) (Vol. 10120, pp. 150–185). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-662-49534-6_5
[13] Falconer, S. M., & Noy, N. F. (2011). Interactive Techniques to Support Ontology Matching. In Z. Bellahsene, A. Bonifati, & E. Rahm (Eds.), Schema Matching and Mapping (pp. 29–51). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-16518-4_2
[14] Fischer, G. (1995). Rethinking and reinventing Artificial Intelligence from the perspective of human-centered computational artifacts. In Lecture Notes in Computer Science (Vol. 991, pp. 1–11). https://doi.org/10.1007/BFb0034793
[15] Ivanova, V., Lambrix, P., & Åberg, J. (2015). Requirements for and Evaluation of User Support for Large-Scale Ontology Alignment. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 9088, pp. 3–20). https://doi.org/10.1007/978-3-319-18818-8_1
[16] Jimenez-Ruiz, E., Grau, B. C., Zhou, Y., & Horrocks, I. (2012). Large-scale Interactive Ontology Matching: Algorithms and Implementation. Frontiers in Artificial Intelligence and Applications, 242, 444 https://doi.org/10.3233/978-1-61499-098-7-444
[17] Khabaza, T. (2014). From Cognitive Science to Data Mining: The First Intelligence Amplifier. In Cognitive Systems Monographs (Vol. 22, pp. 191–204). https://doi.org/10.1007/978-3-319-06614-1_13
[18] Madhavan, J., Bernstein, P. A., Doan, A., & Halevy, A. (2005). Corpus-Based Schema Matching. In 21st International Conference on Data Engineering (ICDE'05) (pp. 57–68). IEEE. https://doi.org/10.1109/ICDE.2005.39
[19] Piest J.P.S., "A Platform Architecture for Industry 4.0 Driven Intelligence Amplification in Logistics," 2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW), Paris, France, 2019, pp. 174-178, https://doi.org/10.1109/EDOCW.2019.00038.
[20] Rahm, E. (2011). Towards Large-Scale Schema and Ontology Matching. In Schema Matching and Mapping (pp. 3–27). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-16518-4_1
[21] eMagiz, https://www.emagiz.com/, 2020
[22] Microsoft Azure Machine Learning, https://azure.microsoft.com/en-gb/services/machine-learning/, 2020