Designing a Hybrid AI Residency

Felipe Leno Da Silva[1], **Silvio Stanzani**[1], **Jefferson Fialho**[1], **Jorge Mondadori**[2], **Muriel Mazzetto**[2], **Felipe Sanches Couto** [2], **Raphael Cobe**[1]
[1]Advanced Institute for Artificial Intelligence (AI2), São Paulo, SP, Brazil
[2]Senai AI Hub, Londrina, PR, Brazil
{felipe.leno, silvio.stanzani, jefferson.fialho, raphael.cobe}@advancedinstitute.ai
{jorge.mondadori, muriel.mazzetto, felipe.couto}@sistemafiep.org.br

Abstract

The industry demand for AI experts raised to unprecedented levels in the last years. However, the increasing demand was not met by the number of skilled professionals in this area. As an effort to mitigate this problem, many companies create AI residency programs to provide in-house practical training. However, we argue that the usual dynamics based on one-on-one mentorship in those programs is very hard to scale and insufficient to meet the demand for AI professionals. In this paper, we describe a hybrid AI residency program that connects educational institutions, partner companies, and prospective residents. This program is designed to be funded by partner companies. Residents are exposed to practical projects of industry interest and are instructed on AI techniques and tools. We describe how we implemented our program, the challenges involved, and the lessons learned after the conclusion of the first residency class. Our program was developed to be inclusive and scalable, and resulted in a high employment rate for our alumni. Furthermore, several partner companies invested in in-house AI teams after the residency, resulting in direct benefits for our local AI community.

Introduction

An "AI residency" is a research training program that teaches promising students (residents) the tools to work on innovative and challenging AI practical projects.

AI Residencies with internal selection and training processes have been implemented by several companies (Google[1], Facebook[2], Microsoft[3], IBM[4], OpenAI[5], etc.) in order to prepare experts to work on their AI projects.

Companies developed those programs for two main reasons: (1) AI experts are a scarce resource, giving rise to a fierce competition for hiring top AI talent, and the consequent rise on their compensation; (2) Industry AI projects often require a practitioner mindset in addition to creativity and scientific knowledge. These soft skills are hard to acquire other than from experience in practical projects. In that scenario, a residency program provides both the opportunity to train and to identify talented candidates that can be hired later if they perform well during the residency.

The current implementation of those programs works well when the company has a vast pool of candidates with specific STEM background. Furthermore, a well-established and active AI department is required so that those professionals can assign projects and provide mentorship. Unfortunately, those candidates and professionals are not available for many companies and for many locations with low STEM-ready talent pool. We argue that the one-on-one mentoring method focused exclusively in projects of interest of a single company cannot scale and is ineffective for leveraging all the potential of an AI Residency.

As an educational institution, we believe that an AI Residency is a very effective tool not only to help with the scarceness of AI talent, but also to provide a vocational practitioner career path for those interested in working with applied AI. Moreover, we also believe that this program has the ability to increase the confidence from companies in AI. In some cases, leading them to invest in in-house R&D AI teams, fostering the development of the AI ecosystem where the residency is implemented.

Therefore, we designed a hybrid AI Residency, where a class of residents without any initial AI expertise is introduced to a set of real local industry demands. Residents have AI classes, focusing on practical methods, tools, and methodologies, while simultaneously working on developing proof of concepts tackling industry demands. We named this program *hybrid AI residency* because it integrates regular classes with project-based learning driven by industry needs. We have designed the classes from scratch, recruited residents, and carried out the program in the cities of Londrina and Curitiba, Brazil, where the first class of 20 residents graduated in 2020.

In this paper, we describe our residency program, the educational challenges we faced, and the lessons learned after our first graduated class. We hope that our paper will encourage new groups to replicate our program in new locations, disseminating AI tools in different communities. We also

[1]https://research.google/careers/ai-residency/

[2]https://research.fb.com/programs/facebook-ai-residency-program/

[3]https://www.microsoft.com/en-us/research/academic-program/microsoft-ai-residency-program/

[4]https://www.research.ibm.com/artificial-intelligence/careers/ai-residency/

[5]https://openai.com/blog/openai-scholars-spring-2020/

expect our program to foster more inclusive training possibilities for AI practitioners, as well as to promote AI interest inside companies.

This paper is organized as follows. We first explain the context in which we built our residency and the motivation behind it. Then, we describe all involved members, their interest, and benefits from the program. After that, we describe in detail how the residency was designed and executed. Finally, we discuss the outcomes from our first class, discuss prospects of scaling-up the program, and give some final remarks about the residency.

Context

Residency programs aim at providing an intensive training through an immersive environment and have been implemented by several companies. Residents learn the basis of AI, apply the knowledge on practical projects, and exercise good practices according to the company's guidelines.

Residencies are usually developed by companies that pair each resident with one full-time employee to act as her/his mentor. The main difference between the residency and a traditional course is that residents will focus almost exclusively in tackling a specific real and practical problem. Therefore, with the help of the mentor, the residents spend their time solving that specific problem and learning the company's practices to deploy solutions into production.

Our motivation for building this program was that we believed that the AI residencies had an yet-unexplored potential of providing a vocational and practical AI instruction. More formal courses such as Ph.D.'s tend to focus on theoretical and/or very specific developments and often provide little applied practice. For this reason, those programs might sound unappealing for people with strong industry mindset. Moreover, those programs alone could not keep up with the demand for AI professionals in the last years.

While the consolidated residencies are working on some of those issues, basing the whole program on one-on-one mentorship makes it very hard to standardize and to scale. Furthermore, those programs have been developed mostly by huge corporations. The interest in AI also exists in smaller companies unable to assign senior professionals to tutor residents.

Therefore, we decided to design a residency program that integrates a more traditional learning paradigm with a project-based industry mindset. Our residency connects multiple interested companies with prospective residents.

Residency Members

We, the Advanced Institute for AI (AI2), have been responsible for hiring instructors and mentors, as well as for preparing all class materials.

In addition to the role taken by us, our residency has three main member roles: the *Residents*, the PCs, and the *Educational Partner* (EP). Each member has their own objectives and expectations for the residency, and they are all essential for the success of the program. Figure 1 illustrates how they interact throughout program.

PCs sponsor the program with financial resources. In return, they specify an AI demand of their interest. The EP recruits PCs and residents. They are also responsible for gathering and filtering the demands from *Partner Companies* (PC). Finally, the *Residents* have two main roles in the program. They take classes for learning all relevant AI concepts and tools. They are also responsible for working on a specific demand submitted by one of the PCs. By the end of the residency, residents are expected to have developed proof of concepts of the demands assigned to them by applying the knowledge and techniques acquired during classes.

Resident

Residents play a central role in the process. Their main objective in joining the program is to develop AI competencies. They received a monthly stipend that corresponds roughly to a Ph.D. scholarship in our location during the whole residency. At the end of the program, all residents that successfully completed all classes and proof of concepts are awarded a certificate of "AI Expert".

Alternatively, it would also be possible to recruit part or all of the residents from the PC staff. However, our PCs also wanted to benefit from the "recruiting" aspect of the residency. Hence the open call for participants was more interesting for them.

Selection Process: For the candidate selection, we opened a public and well-advertised call for participation where the eligibility criteria were:

1. Bachelor degree (or equivalent) in Engineering, Computer Science, Mathematics, Physics, Statistics, other STEM courses or similar Associate Degrees.
2. Basic knowledge of computer programming.

In addition to the required qualifications, we preferred candidates with some previous knowledge of Python programming, data structures, linear algebra, and calculus.

Given the inclusive nature of our call for participants, we received applications from a wide spectrum of candidates, ranging from fresh graduates to experienced software engineers aiming at building AI expertise. The number of applications for our first class was roughly four times the number of available positions. Therefore, we performed a selection process based not only on curriculum evaluation but also on a logic reasoning and programming test. We also asked the applicants record a personal presentation to state their motivation for participating in the program.

We selected 20 residents for the first edition of our hybrid AI residency. Half of the participants were already based in one of the cities where the program was held and 85% of the residents were based in the same state. The remaining 15% of residents relocated from other states in Brazil. The gender ratio was 70% M to 30% F, with most of the residents aged between 24 and 30 years old. We are currently evaluating measures to increase representability, as the pool of applicants was extremely imbalanced.

Partner Companies

PCs have the key role of funding the residency. However, we envision that a similar role could be played by government

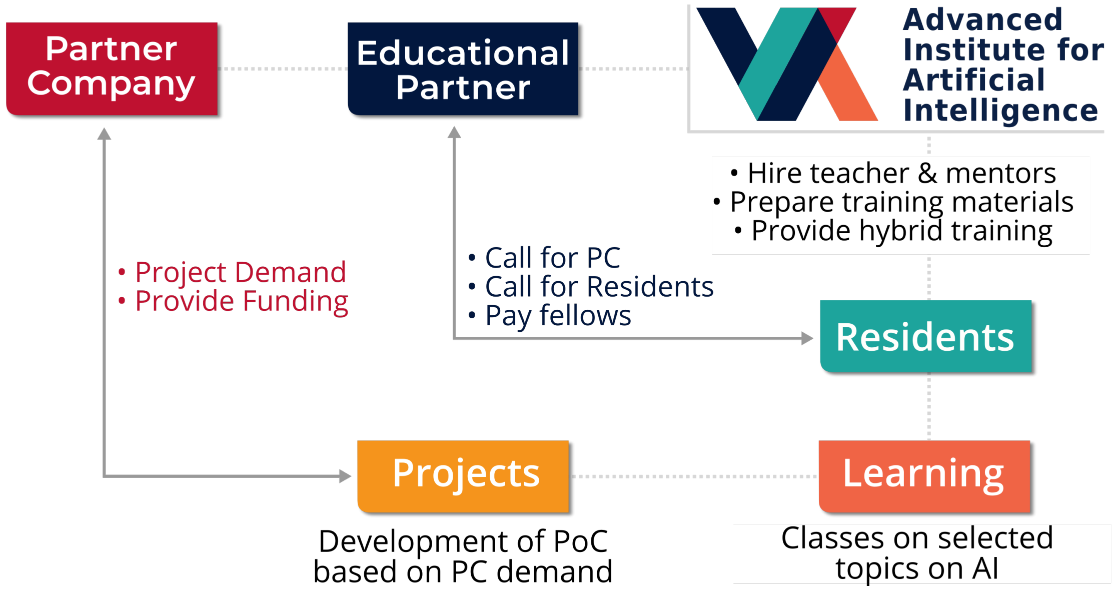

Figure 1: Residency members and their connections. AI2 hires *instructors* and *mentors*. They are responsible for teaching AI classes to residents, while the latter provide mentorship on the project development that the residents have to perform throughout the residency. Partner Companies specify the demand in which they would like to have residents working on, and later receive proof of concepts related to those demands.

funding programs. In general, our PCs are companies interested in leveraging the potential of AI for optimizing their operations or for building new products, but do not have dedicated departments. Each PC is responsible for:

- Describing the demand of interest, providing access to the relevant data;
- Investing funds, that will be managed by the EP to pay resident stipends and build the infrastructure needed.
- Assigning employees that will serve as points of contact between the residents and the company. Those employees will participate in alignment meetings throughout the residency.

In return to their financial investment, a set of residents will work on PC demands for building a proof of concept.

The selection process for PCs is quite similar to the residents'. The EP published a public call for interested companies, where no specific restriction was placed. Additionally, they personally contacted companies that we believed would be interested in the program. Interested PCs ranged from startups to multinational companies.

Each demand was evaluated by the EP to assess whether if they really correspond to an AI project, and in the affirmative case they proceeded to formalize the partnership. The Intellectual Property (IP) rights of the developed proof of concepts are granted to the PC, but it is also of their responsibility to protect those rights submitting patent applications when applicable.

Educational Partner

Together with us, the EP is responsible for connecting the other members and for organizing the residency. Our EP is located in a different city, which further motivated us to carry out the classes (to be described in the next section) in distance education format (Keegan 1996).

While AI2 developed all class materials, and hired instructors and mentors for the residency, the EP performed the above-mentioned selection process for residents and PCs.

All residents have a work station available inside the EP infrastructure. However, since all classes were held remotely, it would also be possible to have residents stationed in the facilities of the PC that submitted the demand they are working in at that point (if they have the space available).

The program management is the main responsibility of the EP together with AI2. We designed the program and made sure that the residents had everything needed to deliver their proof of concepts according to the deadlines (e.g., access to data or domain experts).

Residency Description

Our residency combines a hybrid teaching course with project-oriented learning (Burdewick 2003). Around 30% of the resident's time is dedicated to AI classes, while the remaining 70% of the time is used for developing proof of concepts based on PC demands.

The residency takes one year of full-time dedication (40h/week). The program is composed of four three-month-long cycles named *Iterations* as illustrated in Figure 2. Each iteration is composed of the same phases and characteristics.

The residents are divided into groups before each iteration. Each group will be working on a specific PC demand. At the end of the iteration, all groups are expected to deliver and present a proof of concept. After completing the cycle,

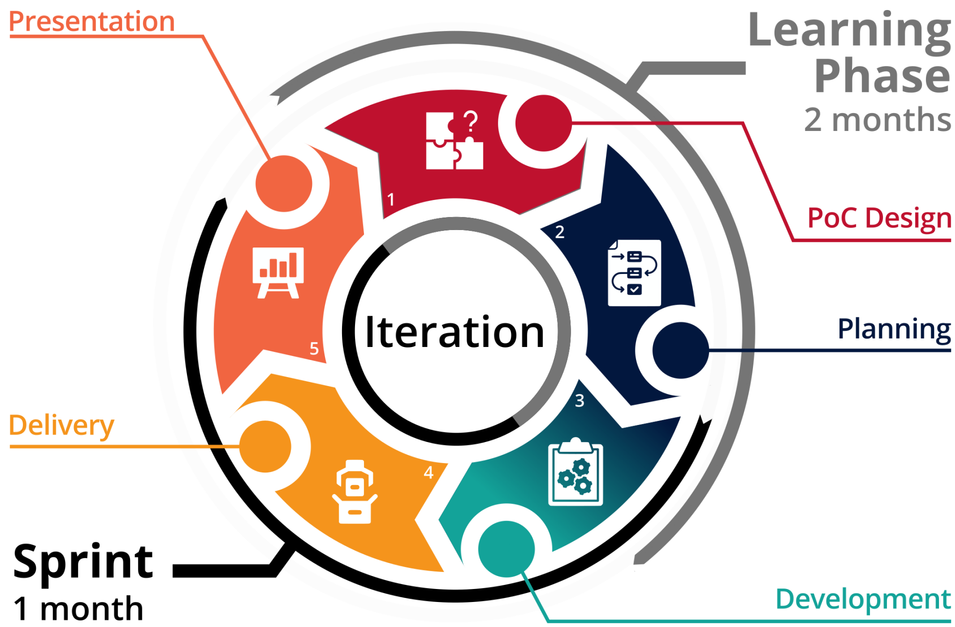

Figure 2: Approximate schedule of the residency. The program is divided into four iterations, each of them composed of a Learning Phase and a Sprint. Simultaneously with the Learning Phase (when the residents have classes), the PoC Design, Planning, and part of the Development project phases are carried out. Then, the rest of Development, Delivery, and Presentation are carried out during the Sprint.

a new group is sorted for the next iteration.

During the first two months of each iteration (the *Learning* phase in Figure 2), instructors from AI2 teach AI classes. During this phase, the residents divide their time between classes and project development. In the last month of the iteration, residents focus exclusively on their projects. During all times, residents have an AI mentor from AI2 to provide guidance on the projects. Typically, PCs proposed one of the following type of demands:

- Problems for which they already had an in-house solution, so that the performance of the AI-based tool could be compared with;
- Automation problems in which PCs think AI could be applied, but that currently are only solved manually; or
- Modules of the PC operation to be optimized through the use of AI.

Learning Phase

We aimed at building an inclusive residency. This means that we focused on recruiting residents that showed potential for applying the knowledge acquired during the program to real world problems, instead of focusing on the ones that already had AI experience.

Therefore, we needed an educational environment where the residents could learn all concepts related to applied AI. This environment was implemented as a *Learning Phase* of two months every three months of residency.

During the learning phases, the residents had classes on AI concepts. Classes were given three times a week, three hours per day in e-learning format (Welsh et al. 2003; Goel and Joyner 2016). All activities were built to work in remote format, so that we would build expertise in tools that scale naturally to new locations and to a larger number of residents.

Unlike regular undergrad (or graduate) AI courses, the residency focuses strongly on preparing the residents to work in industry. Hence, teaching only the AI concepts is insufficient. Thus, divided most classes in a theoretical and a practical component. Whenever possible, classes are divided equally in 1h30 segments of theory and practice.

We also had to consider the diversity of our resident body to prepare the classes. Our residents were a mixture ranging from seasoned software engineers with little maths background to fresh graduates with good theoretical base but very little practical experience. Therefore, we chose to cover all aspects we deemed important for an AI practitioner, resulting in the modules illustrated in Figure 3. Note that we focus extensively on Machine Learning, because the residency primarily prepares for industry and this is currently the area of higher demand. Those are the five different modules:

- *Software Engineering*: Since many of the residents had little to none practical experience with Python, we start with introductory classes on programming and software engineering practical skills. Our classes focuses on teaching tools for building solutions such as *Git* (Spinellis 2012) for code versioning, *Anaconda* (Continuum 2015) for package management, and IDEs such as PyCharm (Islam 2015).
- *Data Science Foundations*: In this module, the residents are taught how to manipulate and to visualize data. In the

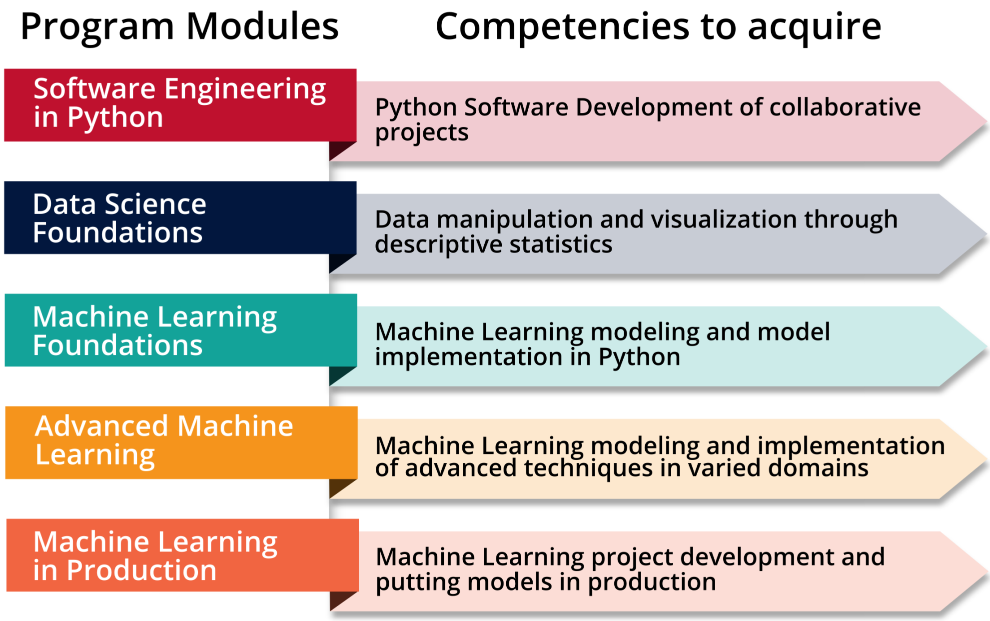

Figure 3: Modules taught during our residency and competencies acquired. Classes start from basic programming and software engineer skills, get through Machine Learning foundations and practical techniques, and finish after we teach tools and techniques for scaling models to production.

theoretical section of the class, we teach the relevant concepts of descriptive statistics so that they can make better use of tools such as Pandas (Wes McKinney 2010) and Matplotlib (Hunter 2007) to manipulate and visualize data.

- *Machine Learning Foundations*: This module teaches the foundation behind basic machine learning algorithms and classical algorithms. By the end of this module, the residents are able to build some of the most classical learning models and to use tools such as scikit-learn (Pedregosa et al. 2011). We found that several EAAI model assignments are quite useful for the practical part of this module (Way et al. 2017).
- *Advanced Machine Learning*: This module focuses on state-of-the-art learning algorithms and Deep Learning approaches. The content in this module is slightly adapted according to the demands submitted by the PC, and is far more practical than the previous ones. By the end of this module, residents are able to implement algorithms using Tensorflow (Abadi et al. 2016) and/or PyTorch (Paszke et al. 2019).
- *Machine Learning in Production*: The last module focuses on scaling up Machine Learning models to production. In addition to learning how to monitor models, residents are taught how to apply containerization, and to use cloud and distributed tools such as Docker (Merkel 2014), MapReduce (Dean and Ghemawat 2008), Spark (Zaharia et al. 2016), and others.

All classes are developed and taught by instructors hired by AI2 following the design described above. We aimed at hiring instructors that were not only knowledgeable in AI but also had teaching experience. The content is reviewed and slightly altered after every new residency, so that the content remains relevant to the current industry market.

Projects

Projects are carried out for building proof of concepts, which allows us to have short project development cycles. Short projects will be enough for the residents to experience a whole cycle of project development, while avoiding taking a large chunk of the residency time in a single project. We leverage the Agile Methodology (Cockburn and Highsmith 2001) to build proof of concepts. We have chosen this methodology because we want to train the residents to develop solutions aligned with the PC demands quickly.

At the beginning of each iteration, the residents are divided into working groups. We aim at providing experience to the residents in different roles. Therefore, each group has its own *project manager* and *developers*. Each resident is assigned to three groups, where they will dedicate roughly one third of their time to the one they are assigned as *project manager* and the remaining of their time to the ones in which they are *developers*. All demands submitted by PCs have a working group assigned to them. Once the demand is matched to a group of residents, a number of activities must be completed during the iteration:

1. *PoC Design*: A kick-off meeting is held with PC representatives to make sure that the demand is well delineated and that all needed data is available to the residents. At the PC's discretion, their representative might be involved in the next phases through periodic alignment meetings. PCs might also opt to only meet again by the end of the iteration.

2. *Planning*: Design and planning of the activities required for delivering a solution. Here, the group of residents has to state clearly how the artifacts produced during this iteration will be delivered (e.g., a Jupyter notebook or an API of interest to the PC). The project manager assign activities and writes a schedule for the rest of the iteration.
3. *Development*: The residents build and evaluate their proposed solution to the problems being tackled during this phase. The project manager oversees the project readjusting the schedule or scope of the solution if needed.
4. *Delivery*: Residents pack everything they have developed and produce a detailed report. All artifacts are made available to the PC at the the end of the iteration.
5. *Presentation*: The results are presented to the PC and all deliverables are made available. A meeting is scheduled to present the results. This delivery marks the end of the iteration.

The *PoC Design* and *Planning* activities take place simultaneously with the learning phase, while the residents alternate between the classes and the planning of their projects. In the last weeks of the learning phase the *Development* activity starts. Then, when the classes come to a end, the residents focus exclusively on the projects for a whole month (sprint). The sprint and the iteration finishes when they delivery a solution to the problem that is being tackled during this iteration. For all project-related activities, the group of residents can count on the mentorship of a mentor.

Teaching and Assessment Strategy

Since residents are not awarded grades as in traditional courses, we need means to assess whether the residents assimilated the content. Given the practical nature of the residency, we chose to assess the residents through small practical exercises that are assigned to them between classes. Those exercises are related to the content already taught, and are assigned to residents after almost all classes.

Whenever possible, we build small challenges for the residents based on the demands and datasets made available by the PCs. Challenges cover the content of several classes, and we propose no more than two during a learning phase. Doing that helps motivating the residents and might produce code useful for their later project development.

We ask residents to present solutions to this challenges and we conduct discussions about the results achieved. This activity is essential for the residents to have the opportunity to deepen their understanding about topics covered by challenge. During the presentation, residents are encouraged to develop a critical view through debates regarding results achieved. This is a excellent opportunity to reinforce part of the content that may have not been understood by everyone.

We also want to develop and improve the ability to build a solution to real practical problems collaboratively. In this sense, we need a feedback about how residents divided the workload for a particular challenge, how much time was spent on each activity, and the quality of the resulting code. For all those challenges, residents are required to use *Git* (Spinellis 2012) for versioning as if it was a real project development, helping them to get familiar with this environment. Moreover, this provided a standard platform for instructors to follow, comment on, and correct group activities.

By tracking the history of each repository, instructors are able to tell apart which part of the code was contributed by each resident. They can also have a sense of how well the lessons were assimilated. As the residents' background varied greatly, the learning experience was different for each resident. Instructors were requested to accompany the residents and to individualize their lessons as much as possible.

Outcomes from our (One-year) Residency

At the end of the program, our perception is that the residency met our expectations. In general, the residents were very dependent on the mentor in the first couple of iterations. In the later iterations, the residents became independent and were able to carry out the projects with little supervision.

The first class of our residency concluded the program in September of 2020. In order to evaluate if the residency was a successful project, we consider the outcome of our first class in three dimensions: (i) the number of residents who were successfully hired to AI-related full-time positions after the residency; (ii) the overall observed impact in the local AI community as a direct result from our residency; (iii) the diversity of machine learning topics investigated on proof of concepts during the course of the residency;

We started the program with 20 residents, from which 18 concluded it successfully. The ones who were not awarded the certificate left the residency because they were hired in AI-related positions before the end of the program. Two months after the program finished, 19 residents are currently employed. While 13 residents are working in AI-related positions, the other six are working in another technical position, such as software engineer or electrical technician. One resident is currently working towards a Ph.D.

The 27 PoC dealed with different topics of the field: temporal series analysis (5), image analysis (5), OCR (4), regression (2), anomaly detection (2), NLP (2), unsupervised classification (1), pattern recognition (1), and chatbot (1). Each resident worked on at least one project from each topic, which provided residents with experience on diverse machine learning topics. For most projects, models were developed and trained from scratch using Keras[6], PyTorch (Paszke et al. 2019), or scikit-learn (Pedregosa et al. 2011).

We also observed indirect benefits of our residency in the local AI ecosystem. From a total of 15 PCs, 8 companies already had at least a small AI team when the program started. At the end of residence, all of the 7 remaining companies decided to invest in AI, creating new jobs that did not exist before. Three of them started an AI area from scratch, while the remaining 4 hired external consultants or companies to transform the PoCs into products.

In summary, we believe that the residency was a successful program in its first edition, which motivated us to start

[6]https://github.com/keras-team/keras

the second class. Despite the strong impact of the COVID-19 pandemic in the local economy, several PCs were still interested in the program.

The second class started in September of 2020 with 13 residents and 10 PCs. Only 1 of the original PCs decided to sponsor the second iteration. A few of the companies reported that they were going through financial hardship, but most of the former PCs reported that they needed some time to transform the delivered PoCs into products, and would go through another round of the residency only after this process is completed. Two of them already signed a contract for the third class in 2021.

Scaling-up the Residency

Three instructors and one mentor were hired for our first residency offering. The instructors divided the classes according to their expertise, hence no classes were taught simultaneously. This proportion worked very well and we predict that one mentor and one instructor would be enough for a class up to 40 residents. The proportion of roughly three residents per PC demand seemed to be ideal, balancing financial feasibility with a reasonably-sized team to build PoCs on time.

We believe that, if this proportion is followed, our residency can be directly scaled into a very big program. If the local language and time zones allow, the program could also be carried out in multiple countries simultaneously, given that the whole program was designed to run remotely.

Conclusion

In this paper, we give a detailed description of an AI residency as we implemented it in Londrina and Curitiba, Brazil. Our residency integrates a hybrid teaching strategy with project-based development of residents. We described all involved members, as well as the process we applied for hiring residents, gathering funds for financing the residency, and attracting Partner Companies. We also describe in this paper our management decisions and the teaching program we defined for the AI classes. We successfully completed the first class of the program, despite the unpredictable challenges brought by the pandemic. We hope that our experience will encourage additional groups to implement similar programs. Our residency was shown to be a resilient and scalable program, and we are planning to expand it to additional locations.

References

Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; et al. 2016. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* abs/1603.04467. URL http://arxiv.org/abs/1603.04467.

Burdewick, I. 2003. Aspects of methodology and education psychology in project-oriented studies. In *International Workshop on Project Oriented Learning*.

Cockburn, A.; and Highsmith, J. 2001. Agile software development, the people factor. *Computer* 34(11): 131–133.

Continuum, A. 2015. Anaconda software distribution. computer software. vers. 2-2.4. 0.

Dean, J.; and Ghemawat, S. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM* 51(1): 107–113. ISSN 0001-0782. doi:10.1145/1327452.1327492.

Goel, A. K.; and Joyner, D. A. 2016. Design of an Online Course on Knowledge-Based AI. In *Symposium on Educational Advances in Artificial Intelligence (EAAI)*, 4089–4094.

Hunter, J. D. 2007. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* 9(3): 90–95. doi:10.1109/MCSE.2007.55.

Islam, Q. N. 2015. *Mastering PyCharm*. Packt Publishing Ltd.

Keegan, D. 1996. *Foundations of distance education*. Psychology Press.

Merkel, D. 2014. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux J.* 2014(239). ISSN 1075-3583.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12: 2825–2830.

Spinellis, D. 2012. Git. *IEEE software* 29(3): 100–101.

Way, T.; Matuszek, P.; Cassel, L.; Papalaskari, M.-A.; and Weiss, C. 2017. Machine Learning for Everyone: Introduction to Classification and Clustering. In *Symposium on Educational Advances in Artificial Intelligence (EAAI)*.

Welsh, E. T.; Wanberg, C. R.; Brown, K. G.; and Simmering, M. J. 2003. E-learning: emerging uses, empirical results and future directions. *international Journal of Training and Development* 7(4): 245–258.

Wes McKinney. 2010. Data Structures for Statistical Computing in Python. In Stéfan van der Walt; and Jarrod Millman, eds., *Proceedings of the 9th Python in Science Conference*, 56 – 61. doi:10.25080/Majora-92bf1922-00a.

Zaharia, M.; Xin, R. S.; Wendell, P.; Das, T.; et al. 2016. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* 59(11): 56–65. ISSN 0001-0782. doi:10.1145/2934664.