Evolution of Artificial Intelligence Programming Languages - a Systematic Literature Review

1,2,3Emmanuel Adetiba, 4Temitope M. Adeyemi-Kayode, 1Adekunle A. Akinrinmade, 1Funmilayo S. Moniuola, 1Oladipupo O. Akintade, 1,2Joke A. Badejo, 4Obiseye O. Obiyemi, 5Surendra Thakur and 4Abdultaofeek Abayomi

1Department of Electrical and Information Engineering, College of Engineering, Covenant University, Ota, Nigeria  
2Covenant Applied Informatics and Communications African Center of Excellence, Covenant University, Ota, Nigeria  
3HRA, Institute for Systems Science, Durban University of Technology, P.O. Box 1334, Durban, South Africa  
4KZN e-Skills CoLab, Durban University of Technology, Durban, South Africa  
5Department of Electrical and Electronic Engineering, Osun State University, Osogbo, Osun State, Nigeria  
6Department of Information and Communication Technology, Mangosuthu University of Technology, P.O. Box 12363 Jacobs, 4026 Durban, South Africa

Abstract: Artificial Intelligence (AI) has received significant attention in recent years. It is being adopted to provide solutions to medicine, engineering, education, government and several other domains. To analyze the state-of-the-art of research in AI, we present a systematic literature review focusing on the Evolution of AI programming languages. Our search returned 7,604 documents; after reviewing these documents, 78 which were relevant for this study were retained. Our research revealed that the prevalence of AI programming language by volume of publications had experienced peaks and valleys between 1963 and 2018; however, between 2015 to 2020, related publications have been experiencing peaks. During the review period, the PROLOG programming language received the most attention in about 49% of publications; this was followed by LISP, which received almost 22%. The remaining attention was shared between Logic and Object-Oriented Programming (LOOP), ARCHLOG, Epistemic Ontology Language with Constraints (EOLC), Python, C++, ADA and JAVA. However, the predominant AI programming language in recent AI software is C/C++, which takes 70% of the modern AI libraries analyzed in this study. Python is used in 60% of the modern AI libraries analyzed. Their prevalence is as a result of their speed, portability and ease of coding, making them effective in developing trending AI libraries such as TensorFlow and Keras.

Keywords: Artificial Intelligence, Programming Language, Python, AI, LISP, PROLOG, JAVA, C++, EOLC, ADA

Introduction

Artificial Intelligence (AI) is concerned with intelligent behaviors in artifacts such as perception, reasoning, learning, communicating and acting in a complex environment. AI involves the science and engineering of machines that possess the listed characteristics, which humans can do, better and faster. The physical symbol system hypothesis states that it has the necessary and sufficient means for general intelligent action. A physical symbol system is a machine like a digital computer capable of manipulating symbolic data—adding numbers, rearranging lists of symbols and replacing some symbols with others (Nilsson, 2010).

Traditionally, computing is used for performing mechanical computations using fixed procedures. Unfortunately, this approach implies complex problems would be more difficult to solve. Another shortcoming is that computers so programmed would have difficulties understanding and adapting to new situations as humans do. AI is different from this traditional approach in that it requires machines to think and tackle such complex assignments. AI was formally coined by John McCarthy in a workshop conducted by IBM at Dartmouth College in 1956 (Nilsson and Nilsson, 1998).

When digital computers were first developed in the 1940s and 1950s, researchers wrote several programs; these programs could play chess, checkers and prove theorems. In the 1960s and 1970s, AI explored various ways to represent problems by developing different search techniques and general heuristics. These enabled the development of
programs used to solve algebraic word problems and symbolic integration. In the 1970 s and 1980 s, as a result of more powerful systems, AI programs were used to build expert systems and by 1997, an IBM program named DEEP BLUE defeated the world chess champion, Garry Kasparov. Interest in AI sagged in the late 1950s. It however, resumed with vigor in the 1980s. Networks of non-linear elements with adjustable-strength interconnections are now recognized as an important class of non-linear modeling tools. AI programming languages are languages capable of implementation of logic (Clark and Gregory, 1986), vastly useful in building expert systems (Hawkinson, 1986) and with features for natural language processing (Weeks and Berghel, 1986). There are many AI programming languages. Some are List Processor (LISP), French for Programming in Logic (PROLOG), Python, JAVA, C++ and ADA.

LISP is a computer programming language with a long history and a distinctive, fully parenthesized prefix notation (Reilly, 2003). It was first conceived in 1958 and it became the programming language of choice for AI research. It was the basis for many ideas in computer science, including tree data structures, automatic storage management, dynamic typing, conditionals, higher-order functions, recursion, the self-hosting compiler (Graham) and the read-eval-print loop (Widemann et al., 2013). Linked lists are LISP’s major data structures and LISP source code is made of lists. Thus, LISP programs can manipulate source code as a data structure, giving rise to the macro systems that allow programmers to create new syntax or new domain-specific languages embedded in LISP.

The idea of PROLOG was first conceived in 1970 and implemented in 1972 by Al quickly Colmerauer, Robert Kowalski and Philippe Roussel (Kowalski, 1988). Since its inception, it was one of the most popular programming languages and has remained the most popular with many variants (Szuba, 1984). Since it is a general-purpose logic programming language, PROLOG is widely used for programming in AI (Clocksin and Mellish, 2012). It differs from other programming languages in that it is intended mainly as a declarative programming language. It has its roots in first-order logic expressed in terms of relations, represented as facts and rules while a computation is initiated by running queries over these relations (Lloyd, 2012). The language is well suited for different branches of AI. For example, logical problems that are randomly selected (Szuba, 1984) (De Raedt et al., 2007); expert systems (Merritt, 2012) and natural language processing (Lally and Fodor, 2011; Pereira and Sieber, 2002).

Python is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. Guido Van Rossum developed it in the early 1990 s (Van Rossum and Drake Jr, 1995). Python’s syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C (Mark, 2007). Python is well adapted for AI tasks (Blank et al., 2003; Laugier and Pertin-Troccaz, 1986), especially in machine learning and natural language processing. It was conceived towards the end of 1980 (Venners, 2003). Still, its implementation began in December 1989 as a successor to the ABC language capable of exception handling and interfacing with the Amoeba operating system (Van Rossum and Drake Jr, 2014). By October 2000, Python 2.0 was released with new features, which included a full garbage collector and support for Unicode (Van Rossum, 2007), versions 2.6 and 2.7 followed while Python 3.0 was released on December 3, 2008 (Blank et al., 2003). Several AI libraries run primarily on the Python infrastructure. For example, Keras, noted for its user-friendliness, modularity and easy extensibility, is a high-level neural network API compatible with Python 2.7-3.6 (Keras, 2018); Theano allows for the evaluation of mathematical expressions using multi-dimensional arrays efficiently (Theano, 2018).

Java was developed in the early 1990 s by James Gosling from Sun Microsystems (McKinley, 2016b). Some of the features of Java are - it can be easily coded and it is highly scalable, making it desirable for AI projects (Rodrigues et al., 2011). It is also portable and can easily be implemented on different platforms since it uses virtual machine technology (Arnold et al., 2005).

C++ was developed in 1979 by Bjarne Stroustrup at Bell Labs and standardized in 1998 as a C language extension. Of all the AI programming languages, it is the fastest and most used by developers in AI projects because of its time sensitivity and also when speed is of higher priority to improve their project execution time (Stroustrup, 2013).

In this study, we carry out a systematic review of published articles on AI programming languages from databases such as SCOPUS, IEEE Xplore and Google scholar. The published articles were reviewed to retrieve the year of execution, development team, capabilities, features and the limitations and applications of AI programming languages.

**Methods**

To provide a systematic review of AI programming languages, we followed the guidelines put forward by (Kitchenham et al., 2004). This systematic review aims to access original articles and full-length review articles that relate to AI programming languages. The process is detailed below:

**Research Questions**

The research questions address forthwith are:

RQ 1: What is the prevalence of AI programming language publications since 1963?  
RQ 2: Which of the AI programming languages have received more attention regarding the volume of research publication produced?
RQ 3: What are the predominant AI programming languages on which recent AI software rely? 
RQ 4: What characteristics of AI programming languages make them suitable or unsuitable for use across platforms?

In addressing RQ 1, we considered the volume of the publications published for each AI programming language since 1963. Each of these programming languages has been described in Table 2 - 10. For RQ 2, we discussed the major AI programming languages; LISP, Prolog, LOOP, ARCHLOG, EOLC, Phyton, C++, ADA and JAVA. We considered the popularity of the languages based on the number of publications that have been published from 1963 to 2020. To answer RQ 3, we analyzed a few recently developed AI software platforms/frameworks and determined the foundational AI programming languages that underlie the frameworks.

Table 1: Search terms on Scopus and IEEE Explore (1963 to 2020)

<table>
<thead>
<tr>
<th>#</th>
<th>Searches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>2</td>
<td>AI</td>
</tr>
<tr>
<td>3</td>
<td>Programming Language</td>
</tr>
<tr>
<td>4</td>
<td>1 OR 2</td>
</tr>
<tr>
<td>5</td>
<td>3 AND 4</td>
</tr>
</tbody>
</table>

Table 2: Extracted studies on LISP artificial intelligence programming languages

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Limitations</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISP</td>
<td>John (1962)</td>
<td>1962</td>
<td>Research Laboratory of Electronics, Massachusetts Institute of Technology, U.S. Army, U.S. Navy and U.S. Air Force</td>
<td>LISP is a mathematical language that uses Symbolic data processing (S-expressions). These S-expressions are stored in a structured list. It performs computational analysis on sequential programs, it has simple internal structure and compatible with other systems.</td>
<td>Ultra slow numerical computation and lack of better representation of block of registers. It has a higher overhead when compared with other conventional programming languages. This has narrowed its use in AI. It is limited with respect to processing and memory requirements. Its size, unwieldiness, &quot;kitchen sink&quot; design strategy and general ADA fixation is disliked by most of its critics. Slower compare to C++</td>
<td>Used in differential and integral calculus, electrical circuit theory, mathematical logic, game playing and other fields of artificial intelligence. A vastly useful tool for the programming of many Expert Systems. It is programming language of choice of most AI researchers. It has been applied in Musical composition and processing foundational language for all programming languages.</td>
</tr>
<tr>
<td>Dixon (1986)</td>
<td>1986</td>
<td>EG&amp;G Idaho Inc, Idaho Falls, ID., USA.</td>
<td>ABle to rapidly analyze large trees by efficiently utilizing a list-based tree structure, search space and rule-based</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kaieler (1986)</td>
<td>1986</td>
<td>Very flexible programming language compared with most others because it is built around a kernel of mathematical principles.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hahn et al. (1988)</td>
<td>1988</td>
<td>Siemens AG, Munich, West Ger, Siemens AG, Munich, West Ger</td>
<td>LISP, a Programming Language and Its Computational Models</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rahn (1990)</td>
<td>1990</td>
<td>The University of Washington</td>
<td>Its features includes, a data structure for music representation, Lisp front (a language for composing music), it is object-oriented and the ability to output Music4P score data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Takeuchi (2002)</td>
<td>2002</td>
<td>Computer Science Department, University of Electro-Communications, Chofu, 182-8585, Japan</td>
<td>Lisp has potentials for systems programming and is suitable for writing operating systems. It also possess capabilities for heterogeneous parallel computing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ellis (2012)</td>
<td>2012</td>
<td>Department of Electrical Engineering and Computer Science, Information and Telecommunication Technology Center, University of Kansas, Lawrence, KS 66045, United States</td>
<td>Coding is fast and efficient due compilers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jaakkola et al. (2019)</td>
<td>2019</td>
<td>Tampere University</td>
<td>Automatic garbage collections was invented for LISP language</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Khomtchouk et al. (2018)</td>
<td>2018</td>
<td>Center for Therapeutic Innovation and Department</td>
<td>Self-modifying codes based on the state of the computer. Ability to provide solutions in terms of algebraic formulas LISP allows programmers to more quickly write programs</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Table 1**: This table lists the search terms used in Scopus and IEEE Explore from 1963 to 2020.
- **Table 2**: This table provides information on studies focusing on LISP artificial intelligence programming languages. It includes the programming language, authors, year, development team, capabilities/features, limitations, and applications.

This comprehensive approach allows for a thorough understanding of the predominant AI programming languages and their characteristics, suitable for use across platforms.
Table 3: Extracted Studies on PROLOG Artificial Intelligence programming language

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Limitations</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROLOG</td>
<td>Clark and Gregory</td>
<td>1986</td>
<td>Dept. of Computing, Imperial College of Science and Technology, University of London</td>
<td>Performs efficient implementation by incorporating parallel mode of computation</td>
<td>The coding of the knowledge and the organization and modularization requires a lot of creativity. Some examples include fact/consequent representation problem and existential quantification problems. One of its drawbacks is the lack of natural mechanism to tackle the issue of uncertainties since by default it is designed to be a two-valued logic programming language. Prolog by default provides limited support for real-life knowledge engineering.</td>
<td>Suitably adapts to applications requiring implementation in logic programming environments. Applied in Octree solid modeling applications. Relational Database applications, natural language processing, theorem proving, automated reasoning. Used for establishing expert systems in specific research activities. Used in logic-circuit model building, expert systems, AI and natural-language interfacing. Used in clinics for detection and classification of QRSs in Electrocardiography (ECG). Forth-based Prolog is used as basis of the expert system component of an autonomous interface for a series of Spacelab experiments It is a well adapted technology for expert systems design Used in real-time expert system to develop autonomous interface for a series of Spacelab vestibular experiments. Used in the probing of a facility’s defenses and to find potential attack paths that meet designated search criteria. Used in electric utility applications, for vol/var dispatch and to increase the capability of electric energy management centers to successfully monitor power system operation and promptly respond to emergencies. Used in representation of a nursing knowledge base. Finds application in HICOM communication system language of choice for knowledge-processing systems suitable for implementing an inference engine for agents. Finds applications in insulin pump systems in hospitals for treatment of diabetic patients. Widely used in artificial intelligence Research Construction of expert systems. Applied in the interpretation of IR spectrums.</td>
</tr>
<tr>
<td>Year</td>
<td>Name</td>
<td>Institution</td>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------------</td>
<td>---------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td>Buzzi</td>
<td>Comparative Physiology and Behavioral Biology Laboratory, Federal Institute of Technology, Zurich, Switzerland</td>
<td>Comparative Physiology and Behavioral Biology Laboratory, Federal Institute of Technology, Zurich, Switzerland.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td>Colmenauer</td>
<td>Faculté des Sciences de Luminy, Unité de recherche associative au CNRS 816, Case 901, 70, route Léon Lachamp, F-13288 Marseille Cedex 9</td>
<td>Faculté des Sciences de Luminy, Unité de recherche associative au CNRS 816, Case 901, 70, route Léon Lachamp, F-13288 Marseille Cedex 9.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td>Patt</td>
<td>Computer Science Division, University of California, Berkeley, CA, United States</td>
<td>Computer Science Division, University of California, Berkeley, CA, United States.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td>Pereira and Calejo</td>
<td>Logic Programming and Artificial Intelligence Group, Universidade Nova de Lisboa (UNL), Monte da Caparica, Portugal</td>
<td>Logic Programming and Artificial Intelligence Group, Universidade Nova de Lisboa (UNL), Monte da Caparica, Portugal.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td>Hayashi et al.</td>
<td>Computer and Network Systems Laboratory, Corporate Research and Development Center, TOSHIBA Corporation</td>
<td>Computer and Network Systems Laboratory, Corporate Research and Development Center, TOSHIBA Corporation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Ellis and Agab</td>
<td>Prolog has a built-in list handling essential in representing tree-based data structures and pattern matching. It can also backtrack automatically.</td>
<td>Prolog has a built-in list handling essential in representing tree-based data structures and pattern matching. It can also backtrack automatically.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Ostermayer et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td>De Raedt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td>Jähkiewicz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td>Nakles</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>Monsieurs et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>Zhang et al.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1987</td>
<td>Martin et al.</td>
<td>Information Technology Research Centre, Department of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, United Kingdom</td>
<td>Information Technology Research Centre, Department of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, United Kingdom.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1987</td>
<td>Postma et al.</td>
<td>Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands</td>
<td>Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1987</td>
<td>Postma et al.</td>
<td>Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands</td>
<td>Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1989</td>
<td>Jaulkolla et al.</td>
<td>Tampere University</td>
<td>Tampere University.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td>Duque-Méndez et al.</td>
<td>Universidad Nacional de Colombia, Manizales, Colombia</td>
<td>Universidad Nacional de Colombia, Manizales, Colombia.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4: Extracted Studies on LOOP Artificial Intelligence programming language

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Limitations</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOOP (Logic and Object-Oriented Programming language)</td>
<td>Suciu et al. (2001)</td>
<td>2001</td>
<td>Dept. of Comput. Sci., Tech. Univ. of Cluj-Napoca, Romania</td>
<td>LOOP extends PROLOG logic programming paradigm with object-oriented features</td>
<td>Lacks mechanisms for structuring knowledge (program clauses)</td>
<td>Used in LP (Linear Programming)-based AI applications</td>
</tr>
</tbody>
</table>

Table 5: Extracted Studies on ARCHLOG Artificial Intelligence programming language

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARCHLOG</td>
<td>Fidjeland and Luk (2006)</td>
<td>2006</td>
<td>Imperial College London, 180 Queen's Gate, London SW7 2AZ, United Kingdom</td>
<td>Can produce high-performance designs without detailed knowledge of hardware development and a framework for designing multiprocessor architectures</td>
<td>Finds application in machine learning and cognitive robotics</td>
</tr>
</tbody>
</table>

Table 6: Extracted Studies on EOLC Artificial Intelligence programming language

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>EOLC (Epistemic Ontology Language with Constraints)</td>
<td>Kumar and Krogh (2007)</td>
<td>2007</td>
<td>Department of Electrical and Computer Engineering, Carnegie Mellon University, Nijmegen, The Netherlands</td>
<td>Used for specifying the epistemic ontology for heterogeneous verification Pittsburg, PA, 15213, USA.</td>
<td>Finds application in redundant flight guidance system and in the heterogeneous Verification of Embedded Control Systems</td>
</tr>
</tbody>
</table>

Table 7: Extracted studies on PYTHON Artificial Intelligence programming languages

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Authors</th>
<th>Year</th>
<th>Development team</th>
<th>Capabilities/features</th>
<th>Limitations</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>PYTHON</td>
<td>McKinley</td>
<td>1991</td>
<td>Department of Analytical Chemistry, Faculty of Science, University of Nijmegen, Toernooiveld, Nijmegen, The Netherlands</td>
<td>Python has improved over a short period of time compare to Java and C++ The efficiency of the programmer is highly improved because of its support for object-oriented design, functional and procedural styles of programming. It has high level syntax. Algorithm can be tested without implementation.</td>
<td>Is not good for mobile computing because of its weak Language for mobile computing The execution is slow in AI development due to the fact that it works with the help of an interpreter unlike C++ and Java</td>
<td>Web content crawling. Developing advanced probabilistic models in AI research</td>
</tr>
<tr>
<td></td>
<td>Bahana et al. (2018)</td>
<td>2018</td>
<td>Computer Science Department, Faculty of Computing and Media, Bina Nusantara University, Indonesia</td>
<td>Contain the utilities; Scrapy, Beautiful Soup and Link Grabber for web crawling, web content extractor and Uniform Resource Locator (URL) extractors, respectively</td>
<td>Beautiful Soup cannot be used as a web crawler by itself</td>
<td>Web content crawling. Developing advanced probabilistic models in AI research</td>
</tr>
</tbody>
</table>

Table 8: Extracted studies on C++ Artificial Intelligence programming languages

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Authors</th>
<th>Year</th>
<th>Capabilities/features</th>
<th>Limitations</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++</td>
<td>Park et al. (2017)</td>
<td>2017</td>
<td>C++ can organize data because it is a multi-paradigm programming that supports object-oriented principles. It has high level of abstraction which makes it good for solving complex problem in AI</td>
<td>The standard library base in C++ is small and even though it has better efficiency of control, big C++ projects are tough to maintain and slow to develop.</td>
<td>Used in virtual robot simulation and synthesis, grasp synthesis, 3D drawing. Any kind of data can be modelled and simulated easily in AI</td>
</tr>
<tr>
<td></td>
<td>McKinley</td>
<td>2016</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Kurniawan et al. (2015)</td>
<td>2015</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ellis and Agah (2012)</td>
<td>2012</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Regarding the characteristics of AI programming languages (RQ 4), we considered the following: Capabilities or features, limitations and application of the AI programming language.

**Search Strategy**

The search for relevant literature was conducted in May 2021 using SCOPUS and IEEE Xplore databases; the dates was set to 1963 to 2020. Furthermore, we extracted more related documents from Google Scholar and reference list. Table 1 shows the search terms used for the databases.

**Selection Criteria**

We searched for studies that are related to AI Programming Language. Specifically, we included studies that revealed the features or capabilities, limitations and...
applications of AI programming languages. We excluded reviews, viewpoints, or editorials. We also reviewed studies that were written in English only.

**Case Definitions**

For this research, we signposted our definition of AI based on the definition put forward by Nils Nilsson (Nilsson, 2010) in his book. “Artificial intelligence is that activity devoted to making machines intelligent and intelligence is that quality that enables an entity to function appropriately and with foresight in its environment.” This definition would guide our final selection of retained studies in subsequent sections.

**Quality Criteria**

We ensured that the AI programming language was explicitly defined and met the case definition for each of the full text accessed as stated in 2.3. Studies with ambiguous details of the features or capabilities, limitations and AI applications were excluded from this review.

**Data Extraction**

Data extraction was conducted by the following reviewers; TJ, AA, FA and OA. The selection of relevant articles was performed by all reviewers. Any disagreement between reviewers on our selection of studies was resolved by the fifth and sixth reviewer EA and JB. From each study, we extracted data on programming language, version, year of execution, development team, capabilities or features, limitations and AI applications. This study seeks to present the evolution of AI programming languages, as such, the retained studies are presented in a chronological order and grouped based on the type of the AI programming language.

**Results**

**Systematic Search**

Our search returned 7,604 documents related to AI programming languages; Scopus (5,818) and IEEE Xplore (435). Furthermore, a thorough manual search was performed on Google scholar and returned 1,039 documents. Hence, a total of 7,604 documents were prepared for analysis. Afterward, 434 duplicated documents were removed. Of the 7,170 documents remaining, 6,665 documents were excluded based on the following criteria; studies that do not relate to Artificial Intelligence programming languages and reviews, books, book chapters, reports, notes, short survey, letter, viewpoints and editorials. A total of 195 full texts and abstracts were assessed for eligibility; after that, 78 studies were retained for qualitative and quantitative synthesis. The highlighted search procedure is graphically represented in Fig. 1.

**Study Characteristics**

The bulk of the retained studies on Artificial Intelligence programming languages can be grouped under the main categories LISP, PROLOG. Two studies in 1986 and 1987 (Futo and Papp, 1986; Martin et al., 1987) respectively dealt with different variants of PROLOG namely TC; PROLOG and FPROLOG. In recent years, we see documentations on new Artificial Intelligence programing language like Logic and Objected-Oriented Programming Language (LOOP) documented by (Suciu et al., 2001) in 2001 which extended PROLOG logic programming language with object oriented features while (Ostermayer et al., 2014) discuss a connection architecture between PROLOG and JAVA. ARCHLOG documented in 2006 (Fidjeland and Luk, 2006) can produce high-performance designs without detailed knowledge of hardware development and a framework for designing multiprocessor architectures; Epistemic Ontology Language with Constraints (EOLC), which is used for specifying the epistemic ontology for heterogeneous verification was documented in 2007 by (Kumar and Krogh, 2007); McKinley described Python in his 2016 paper (McKinley, 2016a), (Ellis and Agah, 2012; Kurniawan et al., 2015; McKinley, 2016a; Park et al., 2017) in 2012,2015,2016 and 2017 respectively discussed the use of C++ in Artificial intelligence programming languages while (Babu et al., 2015; Garg and Kumar, 2017; Kurniawan et al., 2015; McKinley, 2016b; Mittal and Mandalika, 2015; Raff, 2017) discussed the use of JAVA in Artificial Intelligence programming languages.

**Successes, Challenges and Limitations**

**Successes of Artificial Intelligence**

Optimization of AI language has enabled a revolutionary change across many sectors globally. For instance, AI has been deployed across different fields like finance, health, engineering and education. The proliferation of AI languages across these sectors has helped to birth new tech businesses (Rauf and Alanazi 2014). In Education and Research, AI has brought about significant improvement in the quality of delivery of educational resources globally; expert systems have been created to provide seamless learning content to students and researchers across borders (McJones, 2017). In the health sector, AI has been used to assist in automated data management (Giakoumakis and Papakonstantinou, 1987), developing an artificial neural network to assist in rapid patient care (King, 2018). The advancements in AI has provided further dynamics of analysis of face or object recognition techniques for audiovisuals. Also, in music evolution in composing human like notes (Baumann et al., 2002). The invasion of cognitive problem-solving skill has brought innovative ideas that is answering engineering questions (Rauf and Alanazi 2014).
Challenges and Limitations of Artificial Intelligence

The main aim of AI is to build an intelligent machine that will make life easier for humans. The machine should be able to think like humans with some intelligent traits added to it. The programmers want to build some emotional quotient into the machines. Expert development is one of the major problems of AI programming language. LISP; a functional language created as a mathematical notation for computer programmers was developed for lambda calculus, which is not part of undergraduate curriculum in higher institutions.

This makes it more tedious for beginners to master LISP than other object-oriented languages like JAVA. The expert community and library capacity are limited due to this difficulty (Falch and Elster, 2018; McJones, 2017; Priestley, 2017).

Since AI involves building an intelligent machine like humans, it must also face some challenges like humans. Identifying some of these challenges will minimize the associated risks and, at the same time, make sure that we take full advantage of this technology.

Most researchers believe and agree that a super intelligent AI is unlikely to showcase human emotions like love or hate and therefore, it cannot become intentionally benevolent or malevolent. However, the most likely scenario to pose a threat to society is via autonomous weapons. These are weapons that AI systems are programmed to use to kill. If in the hands of the wrong person, these weapons could easily cause mass casualties. This could even lead to an AI war that would also result in mass causalities (Mason, 2010).

Legal challenges related to AI’s application in the financial industry could be related to the consequences of erroneous algorithms and data governance. Due to the lack of appropriate data, erroneous algorithms can leave a big dent in the profits of an organization by making incorrect and perhaps detrimental predictions.
Poor data governance can result in data breaches where customers’ Personal Identifiable Information (PII) that acts as a feedback to an algorithm may get into the hands of hackers and can cause legal challenges for the organization (Heinl, 2014).

Discussion

This study was conducted to clearly understand the evolution of AI programming language from conception to date. The study confirms that John McCarthy pioneered the concept of AI, which has given birth to AI based high-level programming languages such as LISP, PROLOG, etc. Our findings suggest that AI; a multi-disciplinary field, can be highly enriched if further research is geared towards developing more syntax and semantic interaction that could produce robust language understanding systems.

We discuss the answers to our research questions in the following sub-sections.

What is the Prevalence of AI Programming Language Publications Since 1963?

AI programming languages have seen significant research interest since 1963. The research area has seen different peaks and valleys in research outputs. Between 1980 and 1986; a steep increase of 8 to 242 documents in SCOPUS was observed. There is however a significant reduction between 1987 and 2003 before a mild increase to 278 in 2004. Between 2005 and 2014, a dip occurred in research outputs in AI programming language based publications.

Which of the AI Programming Languages Have Received More Attention with Respect to the Volume of Research Publications Being Produced?

The volume of research outputs focusing on the PROLOG programming language has been rather significant. From Table 3, 34 research publications were reviewed for PROLOG in this systematic review, 15 documents were reviewed for the LISP programming language while the remaining 20 documents were reviewed for Logic and OBJECT Oriented Programming (LOOP), ARCHLOG, Epistemic Ontology Language with Constraints (EOLC), Phyton, C++, ADA and JAVA programming languages making it a total of 78 reviewed articles.

What are the Predominant AI Programming Languages on which Recent AI Software’s Rely?

A number of AI libraries have been written based on some of the reviewed foundational AI programming languages, some of these programs are listed in Table 11. As illustrated in the Table, C/C++ has enjoyed the highest level of patronage (N = 7) by developers of modern AI libraries. This is closely followed by Python (N = 6). The reasons for the high adoption of these languages may be attributed to their unique strengths compared with other reviewed languages as earlier illustrated in Table 7 and 8.

What are the Characteristics of AI Programming Languages Which Make Them Suitable or Unsuitable for Use Across Platforms?

One of the most desirable qualities of AI programming languages is speed. PROLOG and LISP are slower compared to C/C++ and Python. An additional ability to support object-oriented principles is desirable. Sometimes it may be necessary to trade between quick response time and execution speed. Other desirable qualities are portability and ease of coding. The combination of C/C++ with Python in developing most of the trending AI libraries (such as Tensor Flow and Keras), as illustrated in Table 11 is born out of the need to complement the weakness of one language with the strength of the other and vice-versa.

Table 11: Details of modern AI libraries and their foundational AI programming languages

<table>
<thead>
<tr>
<th>S/N</th>
<th>Modern AI libraries</th>
<th>Initial release date</th>
<th>Original author</th>
<th>AI programming language platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ALICE</td>
<td>1995</td>
<td>Joseph Weizenbaum</td>
<td>Java</td>
</tr>
<tr>
<td>2</td>
<td>OpenNN</td>
<td>2003</td>
<td>International Center for Numerical Methods in Engineering (CIMNE)</td>
<td>C++</td>
</tr>
<tr>
<td>3</td>
<td>OpenCog</td>
<td></td>
<td></td>
<td>C++, Python</td>
</tr>
<tr>
<td>4</td>
<td>TensorFlow</td>
<td>2015</td>
<td>Google Brain Team</td>
<td>Phython, C++, CUDA</td>
</tr>
<tr>
<td>5</td>
<td>Siri</td>
<td>2011</td>
<td>Apple</td>
<td>Objective C</td>
</tr>
<tr>
<td>6</td>
<td>Neural Designer</td>
<td>2015</td>
<td>Arteinics</td>
<td>C++</td>
</tr>
<tr>
<td>7</td>
<td>Keras</td>
<td>2015</td>
<td></td>
<td>Python</td>
</tr>
<tr>
<td>8</td>
<td>Scikit-learn</td>
<td>2007</td>
<td>David Cournapeau</td>
<td>Python, Cython, C, C++</td>
</tr>
<tr>
<td>9</td>
<td>Pandas</td>
<td>2017</td>
<td>Wes McKinney</td>
<td>Python</td>
</tr>
<tr>
<td>10</td>
<td>SciPy</td>
<td>2017</td>
<td>Travis Oliphant, Pearu Peterson, Eric Jones</td>
<td>Python Fortran, C, C++</td>
</tr>
</tbody>
</table>

1166
Conclusion

This review covers the period between 1963 and 2020, the search criteria on relevant databases was based on the keywords "Artificial Intelligence", "AI" and "programming language". The search returned 7,604 documents, which were narrowed to 78 based on the relevance of information in the publication for our systematic review task. Based on quantitative and qualitative analysis of the resulting documents, we present the successes, challenges and limitations of AI. Our findings revealed that AI programming languages have experienced periods of peaks and valleys from 1963 but have been experiencing peaks since 2015, indicating their prevalence in recent times. Also, in the period of review, PROLOG and LISP have received the most attention. However, in recent AI software, C/C++ and Python are the most deployed owing to their speed, portability and ease of coding. Significant progress has been made in the field that has grown over the last 55 years and practically helped to solve many potential human problems concerning machine vision, gameplaying and automatic natural language processing. This has even stretched beyond the scope of the originators of AI like Turing and McCarthy ever envisaged. More research was geared towards fine tuning the AI like Turing and McCarthy ever envisaged. More

Acknowledgement

This study was carried out at the Advanced Signal Processing and Machine Intelligence Research (ASPMIR) group laboratory, Covenant University, Ota, Nigeria.

KZN e-Skills CoLab, Durban University of Technology, Durban, South Africa provided support for the publication of this work.

Author’s Contributions

Emmanuel Adetiba: Concept development, writing, critical review, final approval.

Temitope M. Adeyemi-Kayode: Writing, critical review, editing.

Adekunle A. Akinrinmade, Funmilayo S. Monimuola and Oladipupo O. Akintade: Writing, editing.

Joke A. Badejo: Editing, critical review of final draft.

Obiseye O. Obiyemi: Critical review of final draft and galley proof.

Surendra Thakur: Critical review of final draft and source for funding of manuscript publication.

Abdultaofeek Abayomi: Editing of the final manuscript and critical review of the final draft.

Ethics

This article is original and the corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References


Clark, K., & Gregory, S. (1986). Parlog: Parallel programming in logic. ACM Transactions on Programming Languages and Systems (TOPLAS), 8(1), 1-49. doi.org/10.1145/5001.5390


Jaśkiewicz, G. (2016). Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots. IEEE Transactions on Computational Intelligence and AI in Games, 8(1), 82-88. doi.org/10.1109/TICAI.2014.2331972


Morozov, A. A., Sushkova, O. S., & Polupanov, A. F. (2017). Towards the distributed logic programming of intelligent visual surveillance applications Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 10062 LNAI, pp. 42-53). doi.org/10.1007/978-3-319-62428-0_4

Nilsson, N. J. (2010). The Quest for Artificial Intelligence: Stanford University. doi.org/10.1017/CBO9780511819346


