Dupa research: Python si C/C++ sunt limbajele cele mai folosite in Machine learning

Programming language for AI-B3-Project

First, let’s look at the overall popularity of machine learning languages. Python leads the pack, with 57% of data scientists and machine learning developers using it and 33% prioritising it for development. Little wonder, given all the evolution in the deep learning Python frameworks over the past 2 years, including the release of TensorFlow and a wide selection of other libraries.  Not only is Python the most widely used language, it is also the primary choice for the majority of its users.

C/C++ is a distant second to Python, both in usage (44%) and prioritisation (19%). Java follows C/C++ very closely, while JavaScript comes fifth in usage, although with a slightly better prioritisation performance than R (7%).

Machine learning scientists working on sentiment analysis prioritise Python (44%) and R (11%) more and JavaScript (2%) and Java (15%) less than developers working on other areas. In contrast, Java is prioritised more by those working on network security / cyber attacks and fraud detection, the two areas where Python is the least prioritised.

**Artificial Intelligence (AI)** in games (29%) and robot locomotion (27%) are the two areas where **C/C++** is favoured the most, given the level of control, high performance and efficiency required. Here a lower level programming language such as C/C++ that comes with highly sophisticated AI libraries is a natural choice

Front-end web developers extend their use of JavaScript to machine learning, 16% prioritising it for that purpose, while staying clear of the cumbersome C/C++ (8%). At the exact opposite stand embedded computing hardware / electronics engineers who go for C/C++ more than others, while avoiding JavaScript, Java and R more than others. Given their investment in mastering C/C++ in their engineering life, it would make no sense to settle for a language that would compromise their level of control over their application. Embedded computing hardware engineers are also the most likely to be working on near-the-hardware machine learning projects, such as IoT edge analytics projects, where hardware may force their language selection.

For **Java**, it’s the front-end desktop application developers who prioritise it more than others (21%), which is also inline with its use mostly in enterprise-focused applications as noted earlier. Enterprise developers tend to use Java in all projects, including machine learning. The company directive in this case is also evident from the third factor that is strongly correlated to language prioritisation — the reason to get into machine learning. Java is prioritised the most (27%) by developers who got into machine learning because their boss or company asked them to. It is the least preferred (14%) by those who got into the field just because they were curious to see what all the fuss was about — Java is not a language that you normally learn just for fun! It is Python that the curious prioritise more than others (38%), another indication that Python is recognised as the main language that one needs to experiment with to find out what machine learning is all about.

**C/C++** is prioritised more by those who want to enhance their existing apps/projects with machine learning (20%) and less by those who hope to build new highly competitive apps based on machine learning (14%). This pattern points again to C/C++ being mostly used in engineering projects and IoT or AR/VR apps, most likely already written in C/C++, to which ML-supported functionality is being added. When building a new app from scratch — especially one using NLP for chatbots — there’s no particular reason to use C/C++, while there are plenty of reasons to opt for languages that offer highly-specialised libraries, such as Python. These languages can more quickly and easily yield highly-performing algorithms that may offer a competitive advantage in new ML-centric apps.

**C++** is a language with a relatively high entry barrier (meaning it’s quite difficult to learn), but the reward is greater because it allows more direct control over the hardware and graphical processes (something very important in video game design). It is an object-oriented language, meaning it uses internal structures to better organize code into reusable blocks (classes and objects).

It is **by far the most commonly used language** for writing game engines, and some engines (such as Unreal) only accept information written in C++. If you only choose one language to learn, this would be it, because most other object-oriented languages stem from C++ (so picking them up will be easier).

**Java** runs on everything, from printers and microwaves to complex video game systems. It is a very dynamic language with lots of applications, making it seem like a good choice to learn. Java is closely related to C++, so learning the two alongside each other would not be difficult.

**Top 16 open source deep learning libraries by Github stars and contributors:**

1. [**TensorFlow**](https://www.tensorflow.org/) was originally developed by researchers and engineers working on the Google Brain Team within Google’s Machine Intelligence research organization. The system is designed to facilitate research in machine learning, and to make it quick and easy to transition from research prototype to production system.  
   Stars: 96655, Contributors: 1432, Commits: 31714, Start: 1-Nov-15. Github URL: [TensorFlow](https://github.com/tensorflow/tensorflow).
2. [**Keras**](https://keras.io/) is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano.  
   Stars: 28385, Contributors: 653, Commits: 4468, Start: 22-Mar-15. Github URL: [Keras](https://github.com/keras-team/keras).
3. [**Caffe**](http://caffe.berkeleyvision.org/) is a deep learning framework made with expression, speed, and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC) and community contributors.  
   Stars: 23750, Contributors: 267, Commits: 4128, Start: 8-Sep-15. Github URL: [Caffe](https://github.com/BVLC/caffe).
4. [**Microsoft Cognitive Toolkit (Previously CNTK)**](https://docs.microsoft.com/en-gb/cognitive-toolkit/) is a unified deep learning toolkit that describes neural networks as a series of computational steps via a directed graph.  
   Stars: 14243, Contributors: 174, Commits: 15613, Start: 27-Jul-14. Github URL: [Microsoft Cognitive Toolkit](https://github.com/Microsoft/CNTK).
5. [**PyTorch**](http://pytorch.org/), Tensors and Dynamic neural networks in Python with strong GPU acceleration.  
   Stars: 14101, Contributors: 601, Commits: 10733, Start: 22-Jan-12. Github URL: [PyTorch](https://github.com/pytorch/pytorch).
6. [**Apache MXnet**](https://mxnet.incubator.apache.org/) is a deep learning framework designed for both efficiency and flexibility. It allows you to mix symbolic and imperative programming to maximize efficiency and productivity.  
   Stars: 13699, Contributors: 516, Commits: 6953, Start: 26-Apr-15. Github URL: [Apache MXnet](https://github.com/apache/incubator-mxnet).
7. [**DeepLearning4J**](https://deeplearning4j.org/) is part of the Skymind Intelligence Layer, along with ND4J, DataVec, Arbiter and RL4J. It is an Apache 2.0-licensed, open-source, distributed neural net library written in Java and Scala.  
   Stars: 8725, Contributors: 141, Commits: 9647, Start: 24-Nov-13. Github URL: [DeepLearning4J](https://github.com/deeplearning4j/deeplearning4j).
8. [**Theano**](http://deeplearning.net/software/theano/) allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently. However, in September 2017, Theano announced that any further major developments would cease after the 1.0 release. Don’t let this put you off though, it is still an extremely powerful library that you can carry out deep learning research with it at any time.  
   Stars: 8141, Contributors: 329, Commits: 27974, Start: 6-Jan-08. Github URL: [Theano](https://github.com/Theano/Theano).
9. [**TFLearn**](http://tflearn.org/) is a modular and transparent deep learning library built on top of TensorFlow. It was designed to provide a higher-level API to TensorFlow in order to facilitate and speed-up experiments, while remaining fully transparent and compatible with it.  
   Stars: 7933, Contributors: 111, Commits: 589, Start: 27-Mar-16. Github URL: [TFLearn](https://github.com/tflearn/tflearn).
10. [**Torch**](http://torch.ch/) is the main package in Torch7 where data structures for multi-dimensional tensors and mathematical operations over these are defined. Additionally, it provides many utilities for accessing files, serializing objects of arbitrary types and other useful utilities.  
    Stars: 7834, Contributors: 133, Commits: 1335, Start: 22-Jan-12. Github URL: [Torch](https://github.com/torch/torch7).
11. [**Caffe2**](https://caffe2.ai/) is a lightweight, modular, and scalable deep learning framework. Building on the original Caffe, Caffe2 is designed with expression, speed, and modularity in mind.  
    Stars: 7813, Contributors: 187, Commits: 3678, 21-Jun-15. Github URL: [Caffe2](https://github.com/caffe2/caffe2).
12. [**PaddlePaddle**](http://www.paddlepaddle.org/) (PArallel Distributed Deep LEarning) is an easy-to-use, efficient, flexible and scalable deep learning platform, which is originally developed by Baidu scientists and engineers for the purpose of applying deep learning to many products at Baidu.  
    Stars: 6726, Contributors: 120, Commits: 13733, 28-Aug-16. Github URL: [PaddlePaddle](https://github.com/PaddlePaddle/Paddle).
13. [**DLib**](http://dlib.net/) is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems.  
    Stars: 4676, Contributors: 107, Commits: 7276, Start: 27-Apr-08. Github URL: [DLib](https://github.com/davisking/dlib).
14. [**Chainer**](http://chainer.org/) is a Python-based, standalone open source framework for deep learning models. Chainer provides a flexible, intuitive, and high performance means of implementing a full range of deep learning models, including state-of-the-art models such as recurrent neural networks and variational auto-encoders.  
    Stars: 3685, Contributors: 160, Commits: 13700, Start: 12-Apr-15. Github URL: [Chainer](https://github.com/pfnet/chainer).
15. [**Neon**](http://neon.nervanasys.com/) is Nervana's Python-based deep learning library. It provides ease of use while delivering the highest performance.  
    Stars: 3466, Contributors: 77, Commits: 1112, Start: 3-May-15. Github URL: [Neon](https://github.com/NervanaSystems/neon).
16. [**Lasagne**](http://lasagne.readthedocs.org/) is a lightweight library to build and train neural networks in Theano.  
    Stars: 3417, Contributors: 64, Commits: 1150, Start: 7-Sep-14. Github URL: [Lasagne](https://github.com/Lasagne/Lasagne).

Bibliography:

* <https://www.gamedesigning.org/career/programming-languages/>
* <https://towardsdatascience.com/what-is-the-best-programming-language-for-machine-learning-a745c156d6b7>
* <https://railsware.com/blog/2018/08/09/best-libraries-and-tools-to-start-off-with-machine-learning-and-ai/>
* <https://www.kdnuggets.com/2018/04/top-16-open-source-deep-learning-libraries.html>