Pierre Wilmot

pierre.wilmot@gmail.com
pierre-wilmot.github.io/resume/

PROFILE

I'm a software engineer with a background in video game development. I now have 10 years of programming experience, with skills ranging from low level optimisation to software architecture. I'm now doing research in Deep Learning, with a special interest for creative application of machine learning. I'm also interested in Cryptocurrencies and blockchain technologies.

EXPERIENCE

Machine Learning Engineer, Fetch.ai, Cambridge, UK 2019 - Present
Designing, implementing and optimising the machine learning framework for the Fetch virtual machine. Re-implemented Word2Vec in C++ as a compute graph in order to build a semantic search engine for the Fetch Open Economic Framework.
— C++ / GTest /Python

Software Engineer, Deutsche Börse, Prague, Czech Republic 2018
Developing and refactoring the EUREX exchange codebase. Leading the implementation of the new TRF Baskets feature. Worked on decoupling the application logic from the messaging and database components in order to be able to set up proper unit testing and run the application in isolation from the rest of the software ecosystem. This led to much more robust and easier to understand code and faster development times.
— C++ / Boost Unit Test / MySQL

Deep Learning Engineer, Artomatix, Dublin, Ireland 2016 - 2018
Working mainly on parametric neural texture synthesis and style transfer. Achieved state of the art results in that domain and published a paper called "Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses". Also worked Neural texture Weathering, texture Segmentation and Single Image Super Resolution. Wrote a neural network runtime from scratch using C# and OpenCL for integration with the company product as well as various GPU modules to speed up deep learning research.
— C++ / C# / OpenCL / Cuda / Python / Lua / (Py)Torch / Tensorflow

Software Engineer, Ubisoft, Craiova, Romania 2015
Setting up the mobile build system to build and run unit test on mobile devices (iOS and Android). Also working as part of the Ubiservices group to develop a cross-platform library providing online features for all Ubisoft games.
— C++ / Python / iOS / Android

Infrastructure Engineer, Blackmagic Design, Melbourne, Australia 2014
Setting up the build system (Python) to be able to request a build through a web UI in one click. The system was checking out the codebase from a git repository, building it on various platforms, testing it, and packaging it as a ready to ship package.
— Python / Git

Video Game Programmer, Good Game Productions, Melbourne, Australia 2014
Porting the in house game creation framework from Cocos2d (Objective-C) to Cocos2d-x (C++). The framework allowed game designer to set-up a scene layout in a drag-and-drop editor and to export everything as a scene file ready to load in the game engine for faster content creation.
— C++ / Objective-C / iOS / Android / Cocos2d-x

Junior developer, OutOfTheBit, London, UK 2013
Mobile game development using an in-house cross-platform framework (C++). Created 3 games in one year (Four in a row / Dots & Boxes / Sound Ride), which where published for iOS, Android and Windows Phone. All projects were developed by a team composed of a developer and a designer.
— C++ / iOS

EDUCATION

Epitech, Lille, France — 2010 - 2012
Deep Learning course by Yann LeCun, Collège de France, Paris, France — 2016

SKILLS

C++ / C / OpenCL / CUDA / Python / Lua / (Py)Torch / MySQL / Git / CMake

PUBLICATION

Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses, Pierre Wilmot, Eric Risser, Connelly Barnes — 2017

PERSONAL PROJECTS

- Mobile games : Laser Maze, 2048 & Snake, published under the brand Owl Games. Development in C++ using cross platform framework cocos2d-x for iOS and Android. Game also include a reporting module connecting to a web server to track events and volume of users.
- Neural network inference runtime. Written in C++ and OpenCL. Support import from Torch and Tensorflow saved format. Implement the basic building blocks of convolutional networks using high performance GPU matrix multiplication provided by clBLAS. Provide ability to run networks like VGG and style transfer networks as well as texture segmentation and image based search demo programs.
- kraken.com client, written in C++. Allow to pull ticker information and user balances from the exchange API. Also include a GUI and a API functionality to interface with a python client in order to run deep learning experiments. C++ code runs as a service that pulls data and keeps a history of recent values. Python client retrieve data through a shared memory segment in order to train a trading agent through reinforcement learning.