



Machine Learning 02 : From Paper to Product

SLQ Wiki Fabrication Lab 2024/11/04 00:50

Machine Learning 02 : From Paper to Product

With State Library closed to the public due to COVID-19, for this Machine Learning (ML) workshop we won't be able to use the Digital Media Lab at The Edge. Instead will be exploring examples of how a Machine Learning idea becomes a product that we can use, following projects from a research paper through to commercial software product.

In the second half of the workshop, we will take a look at Google Colab - a free ML toolkit that we can test out online. The workshop is not an introduction to coding or math, but we will give a of general overview of how ML is defined, how it is developed into a product and where it is commonly used today.

Requirements

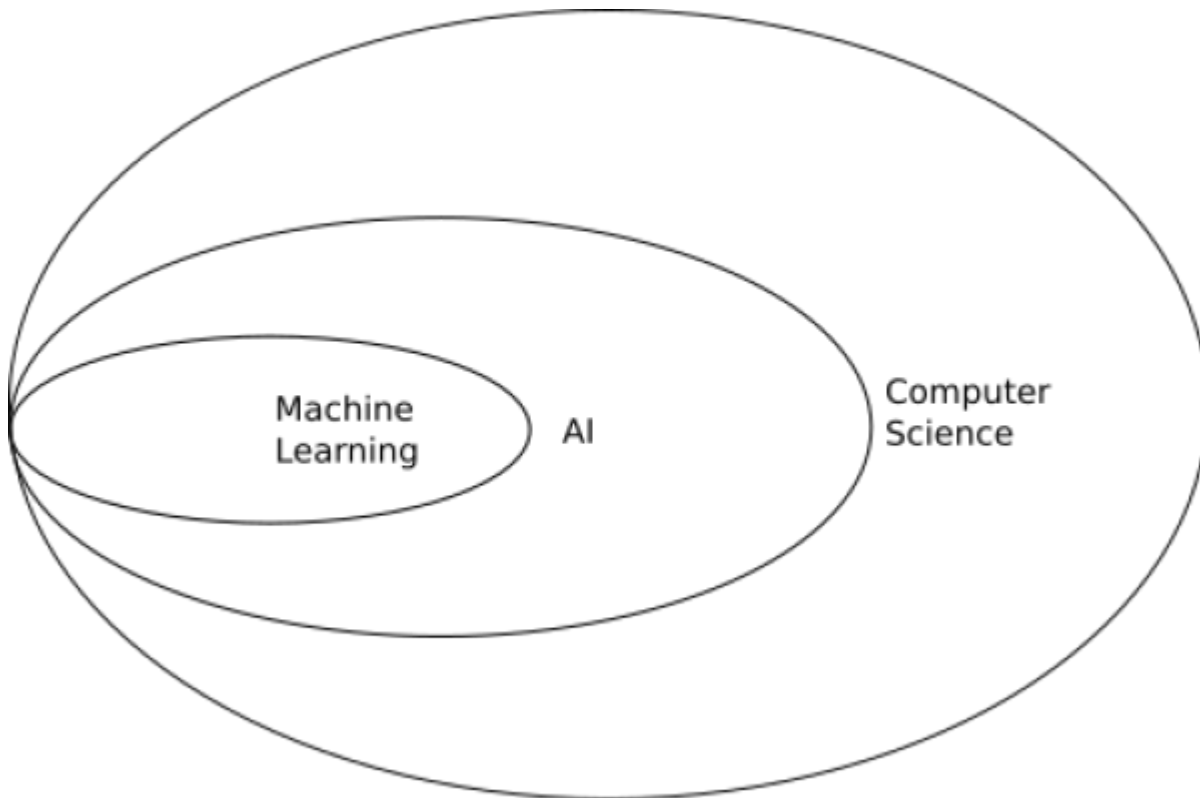
All we need to get started for this workshop is a Google account to access Google Colab in the second half of the workshop. If you don't have one you can quickly [sign up](#). If you don't want to create a Google account, you can always just follow along with the examples.

Outcomes

- A basic Machine Learning background
- Short history of speech synthesis
- ML speech synthesis examples:
 - Commerical
 - Open Sourc research
- using Google Colab
- using Spleter (audio source separation)

Background

Machine Learning(ML) is a subset of Artificial Intelligence (AI) which is is a fast moving field of computer science (CS). A good way to think about how these fields overlap is with a diagram.



Machine Learning - Why Now?

While many of the concepts are decades old, and the mathematical underpinnings have been around for centuries, the explosion in use and development of ML learning has been enabled by the creation and commercialisation of massively parallel processors. This specialised computer hardware most commonly found in Graphics Processing Units (GPUs) inside desktop and laptop computers and takes care of the display of 2D and 3D graphics. The same processing architecture that accelerates the rendering of 3D models onscreen is ideally suited to solve ML problems, resulting in specialised programming platforms, Application Programming Interfaces (APIs) and programming libraries for AI and ML.

Common Machine Learning Uses

One way to think of ML is as a *recommendation system*.

Based on input data (a lot of input data¹⁾)

- a machine learning system is trained
- a model is generated
- the model is used can make recommendations (is implimented) on *new* data.

:workshops:prototypes:machine_learning:ideepcolor:8725096366_d1fe677cc5_o.jpg

One extremely common application of this is image recognition.



:workshops:prototypes:machine_learning:ideepcolor:2020-02-21_14_19_57-8725096366_d1fe677cc5_o_fb.jpg

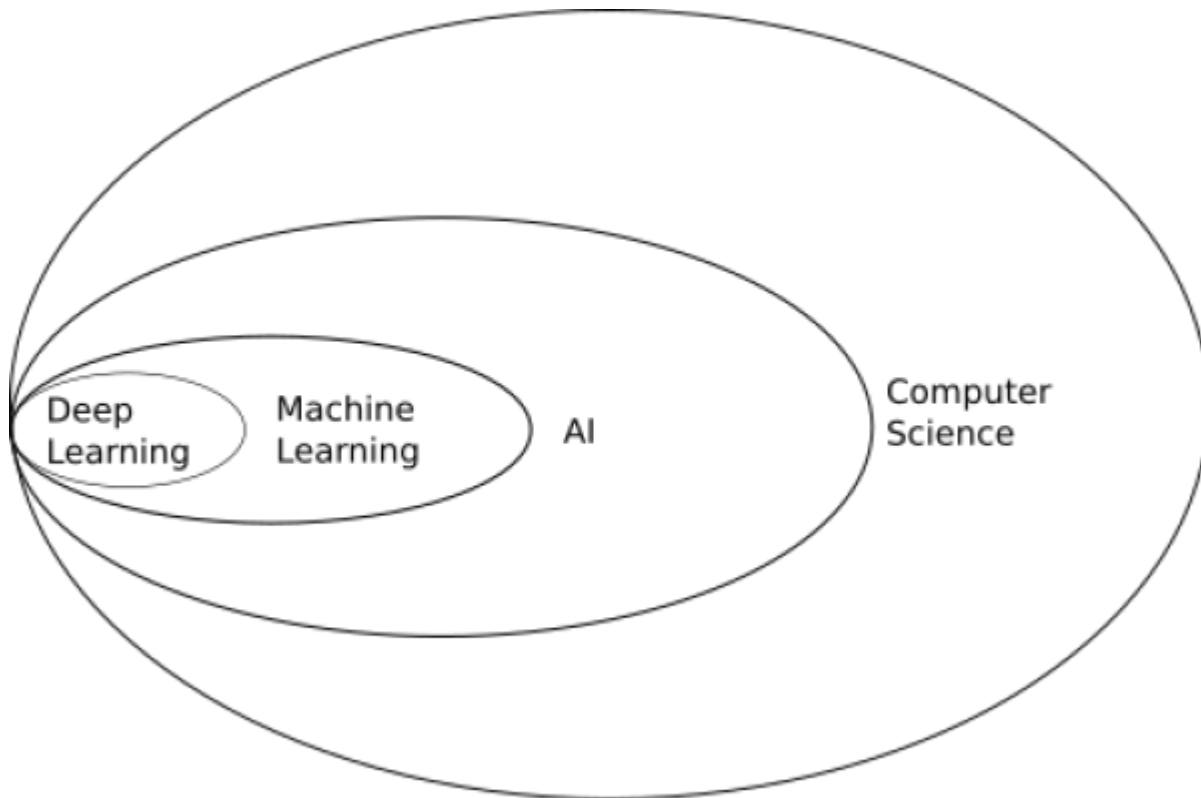
When facebook asks you to tag a photo with names, you are providing them with a nicely annotated data set for **supervised learning**. They can then use this data set to train a model that then recognises (makes a recommendation) about other photos **with you or your friend in it**.

Snapchat filters use image recognition to make a map of your features, then applies masks and transformations in real-time.



Deep Learning

Today we are going to go a little “deeper” inside ML, exploring deep learning. Deep learning used multiple layers of algorithms, in an artificial neural network, inspired by the way the human neural networks inside all of us.



ML - From Paper to Product

We'll be exploring a few ML ideas, but to start with let's follow "[Real-Time User-Guided Image Colorization with Learned Deep Priors](#)", by Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S. Lin, Tianhe Yu, Alexei A. Efros from paper to product. It's not a recent paper in ML terms where it seems every month brings another breakthrough, but we can follow this paper right through to its release in [Adobe Photoshop Elements 2020](#).

Research Papers on arXiv.org

[arXiv.org](#) is probably the world's biggest and fastest growing collection of preprint electronic scientific papers from mathematics, physics, astronomy, electrical engineering, computer science, quantitative biology, statistics, mathematical finance and economics. All of the ML ideas we will be looking at are either first published on arXiv.org, or reference papers on the site.

Finding a Paper

Papers on arXiv.org are moderated but not peer-reviewed, which means the speed and volume of publishing on this open-access repository is overwhelming. But to get started, let's say we are interested in re-colourisation of black and white images of our grandparents at their wedding, but we don't want to do all the work ourselves. We'd also like to interact in real-time and guide the process, so we can get the colour of granddad's suit and grandma's bouquet just right.

So lets search for “real-time guided image colourisation” but we'll use the American English spelling “colorization”. [Searching](#) for “real-time guided image colorization” brings up our paper straight away, with handy [PDF link](#)²⁾.

The screenshot shows a search engine interface. At the top, it says "Showing 1-1 of 1 results for title: real-time guided image colorization". Below this is a search bar containing the text "real-time guided image colorization". To the right of the search bar are options for "Title" and a "Search" button. Below the search bar, there are options to "Show abstracts" (selected) and "Hide abstracts". To the right, there is a "Feedback?" link and an "Advanced Search" link. Below the search bar, there are options for "50 results per page" and "Sort results by Announcement date (newest first)" with a "Go" button. The first result is listed as "1. arXiv:1705.02999 [pdf, other] cs.CV GR". The title of the paper is "Real-Time User-Guided Image Colorization with Learned Deep Priors". The authors are listed as Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S. Lin, Tianhe Yu, and Alexei A. Efros. The abstract text reads: "Abstract: We propose a deep learning approach for user-guided image colorization. The system directly maps a grayscale image, along with sparse, local user 'hints' to an output colorization with a Convolutional Neural Network (CNN). Rather than using hand-defined rules, the network propagates user edits by fusing low-level cues along with high-level semantic information, learned from large-scale data. We tr... More". The submission date is "Submitted 8 May, 2017; originally announced May 2017." and the comments are "Comments: Accepted to SIGGRAPH 2017. Project page: https://richzhang.github.io/ideepcolor".

Examining the Abstract

All research papers begin with an abstract, and a well written abstract will tell us all we need to know about whether the paper is relevant for you, particularly if we are looking for working demonstration. This time we are in luck - there is a link to a [an ideepcolor demo site](#) at the end of the abstract.

Check out the Demo

The demo for ideepcolor looks great and we've got a link at the top of the page, where ideepcolor is [implemented](#) on github.

Code Implementation on github.com

[Github.com](#) is a website used by software developers to create, collaborate and share source code, and is most likely the largest repository of source code in the world. Github is named after [git](#), a free and open-source(FOSS) distributed version-control system for tracking changes in source code during software development. Git means that developers from all over the world can work on the same code and if the project is open source, build on, expand and re-purpose shared codes³⁾. But lets back up a bit and cover off on what source code is.

Using the Source

[source code](#) is the instructions for a computer program contained in a simple text document.

For a computer to run a program, the source code either has to be

***compiled** into binary machine code by a [compiler](#):

- his file is executable - in this case execute just means can be read, understood and acted on by the computer, or
- **interpreted** by another program, which directly executes the code

Here is a example of source code. in this case its a simple program in the C programming language that shows on the screen "Hello, World"

```
#include <stdio.h>

int main(void)
{
    printf("Hello, world!\n");
    return 0;
}
```

Despite the strange symbols, if you know how the C language is written, this program is **human readable**.

Once this code is run through a compiler, we get a binary executable file - which is **machine readable**.

But with the right tools (like a HEX editor) we can still open the file and edit it.

Here is the binary for our "Hello World!" program.

The image shows a terminal window titled "Terminal - ccc@ccc-server: ~/helloworld". The terminal displays a hex dump of the word "hello". The address ranges from 3E8 to 7A8. The hex values are shown in columns, and the ASCII characters are shown in the right column. The word "hello" is clearly visible in the ASCII column.

```

3E8  0c 20 00 48 85 c0 74 05 e8 3b 00 00 00 48 83 c4 08 c3 00 00 00 00 00 00  .H.t.;...H.....
400  ff 35 02 0c 20 00 ff 25 04 0c 20 00 0f 1f 40 00 ff 25 02 0c 20 00 68 00  .5...%...@.%..h.
418  00 00 00 e9 e0 ff ff ff ff 25 fa 0b 20 00 68 01 00 00 00 e9 d0 ff ff ff  .....%...h.....
430  ff 25 f2 0b 20 00 68 02 00 00 00 e9 c0 ff ff ff 31 ed 49 89 d1 5e 48 89  .%.h.....l.I.^H.
448  e2 48 83 e4 f0 50 54 49 c7 c0 c0 05 40 00 48 c7 c1 50 05 40 00 48 c7 c7  .H...PTI...@.H..P.@.H..
460  2d 05 40 00 e8 b7 ff ff ff f4 66 0f 1f 44 00 00 b8 47 10 60 00 55 48 2d  -.@.....f..D..G.`UH-
478  40 10 60 00 48 83 f8 0e 48 89 e5 77 02 5d c3 b8 00 00 00 00 48 85 c0 74  @.`H...H..w.).....H.t
490  f4 5d bf 40 10 60 00 ff e0 0f 1f 80 00 00 00 00 b8 40 10 60 00 55 48 2d  .].@.`.....@.`UH-
4A8  40 10 60 00 48 c1 f8 03 48 89 e5 48 89 c2 48 c1 ea 3f 48 01 d0 48 d1 f8  @.`H...H..H..H..?H..
4C0  75 02 5d c3 ba 00 00 00 00 48 85 d2 74 f4 5d 48 89 c6 bf 40 10 60 00 ff  u.].....H..t.]H..@.`..
4D8  e2 0f 1f 80 00 00 00 00 80 3d 59 0b 20 00 00 75 11 55 48 89 e5 e8 7e ff  .....=Y. .u.UH..~.
4F0  ff ff 5d c6 05 46 0b 20 00 01 f3 c3 0f 1f 40 00 48 83 3d 18 09 20 00 00  .].]F.....@.H.=...
508  74 1e b8 00 00 00 00 48 85 c0 74 14 55 bf 20 0e 60 00 48 89 e5 ff d0 5d  t.....H..t.U. .H....]
520  e9 7b ff ff ff 0f 1f 00 e9 73 ff ff ff 55 48 89 e5 bf d4 05 40 00 b8 00  .{.....s...UH.....@...
538  00 00 00 e8 d0 fe ff ff 5d c3 66 2e 0f 1f 84 00 00 00 00 0f 1f 40 00  .....].f.....@.
550  41 57 41 89 ff 41 56 49 89 f6 41 55 49 89 d5 41 54 4c 8d 25 a8 08 20 00  AWA..AVI..AUI..ATL.%...
568  55 48 8d 2d a8 08 20 00 53 4c 29 e5 31 db 48 c1 fd 0c 48 83 ec 08 e8 5d  UH.-..(SL).l.H...H....]
580  fe ff ff 48 85 ed 74 1e 0f 1f 84 00 00 00 00 00 4c 89 ea 4c 89 f6 44 89  !.H..t.....L..L.D.
598  ff 41 ff 14 dc 48 83 c3 01 48 39 eb 75 ea 48 83 c4 08 5b 5d 41 5c 41 5d  .A..H...H9.u.H...[A\A]
5B0  41 5e 41 5f c3 66 66 2e 0f 1f 84 00 00 00 00 00 f3 c3 00 00 48 83 ec 08  A^A_.ff.....H...
5C8  48 83 c4 08 c3 00 00 00 01 00 02 00 48 65 6c 6c 6f 2c 20 57 6f 72 6c 64  H.....Hello, World
5E0  21 2f 6e 00 01 1b 03 3b 30 00 00 00 05 00 00 00 1c fe ff ff 7c 00 00 00  !/n.....;0.....|...
5F8  5c fe ff ff 4c 00 00 00 49 ff ff ff a4 00 00 00 6c ff ff ff c4 00 00 00  \...L...I.....l.....
610  dc ff ff ff 0c 01 00 00 14 00 00 00 00 00 00 00 01 7a 52 00 01 78 10 01  .....zR..X..
628  1b 0c 07 08 90 01 07 10 14 00 00 00 1c 00 00 00 08 fe ff ff 2a 00 00 00  .....*...
640  00 00 00 00 00 00 00 00 14 00 00 00 00 00 00 01 7a 52 00 01 78 10 01  .....zR..X..
658  1b 0c 07 08 90 01 00 00 24 00 00 00 1c 00 00 00 98 fd ff ff 40 00 00 00  .....$......@...
670  00 0e 10 46 0e 18 4a 0f 0b 77 08 80 00 3f 1a 3b 2a 33 24 22 00 00 00 00  ...F..J..w...?;*3$"...
688  1c 00 00 00 44 00 00 00 9d fe ff ff 15 00 00 00 00 41 0e 10 86 02 43 0d  ..D.....D.....A....C...
6A0  06 50 0c 07 08 00 00 00 44 00 00 00 64 00 00 00 a0 fe ff ff 65 00 00 00  .P.....D...d.....E....
6B8  00 42 0e 10 8f 02 45 0e 18 8e 03 45 0e 20 8d 04 45 0e 28 8c 05 48 0e 30  .B....E....E..E.(.H.0
6D0  86 06 48 0e 38 83 07 4d 0e 40 6c 0e 38 41 0e 30 41 0e 28 42 0e 20 42 0e  .H.8..M.@l.8A.0A.(B. B.
6E8  18 42 0e 10 42 0e 08 00 14 00 00 00 ac 00 00 00 c8 fe ff ff 02 00 00 00  .B..B.....
700  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
718  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
730  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
748  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
760  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
778  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
790  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
7A8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....

```

Open Source

Dokuwiki (the software we are using for this wiki) is open source, and developed publicly, and freely [available](#) on the internet. Anyone is able to grab the source code and run it, modify it or redistribute it.

Below is an example of the **open source** code for this wiki, which is written in a language called [php](#).

```
// define all DokuWiki globals here (needed within test requests but also helps to keep track)
global $ACT, $INPUT, $QUERY, $ID, $REV, $DATE_AT, $IDX,
$DATE, $RANGE, $HIGH, $TEXT, $PRE, $SUF, $SUM, $INFO, $JSINFO;
if(isset($_SERVER['HTTP_X_DOKUWIKI_DO'])) {
$ACT = trim(strtolower($_SERVER['HTTP_X_DOKUWIKI_DO']));
} elseif(!empty($_REQUEST['idx'])) {
```

```
$ACT = 'index';  
} elseif(isset($_REQUEST['do'])) {  
$ACT = $_REQUEST['do'];  
} else {  
$ACT = 'show';  
}
```

How did we get hold of the source code for this wiki? In this case all we did was look in the dokuwiki source found on [github](#) pick bit of code at random and throw it in our wiki.

So, finding the source for open software is easy. but to do the same thing with closed source program is usually difficult or impossible. Either you purchase or are given access to the code. Any other method may break all manner of licenses and laws.

ideepcolor on Github

For a project like ideepcolor, Github is where researchers and developers describe how they achieved their results with real, working code. There is generally an introduction, which should contain any major updates to the project, then we go through the prerequisites, setting up or getting started, installation, training and (hopefully) application. There are number of ways to demonstrate application of a model, ideepcolor has built a custom Graphical User Interface (GUI), which we demonstrated this project in our first ML workshop. More commonly demos are done with a [jupyter notebook](#) or [Google Colab](#) notebook. Now, lets look at the updates at the top of the ideepcolor repository - which tell us:

10/3/2019 Update: Our technology is also now available in Adobe Photoshop Elements 2020. See this [blog](#) and [video](#) for more details.

So it looks like this project has moved to the next stage - integrating ideepcolor into a commercial product.

Comercialisation: Standalone or Software as a Service

Many ML projects stay as research only, but for those that make it into commercial production, there is generally two paths:

- As a component of an existing software project as illustrated by ideepcolor.
- Offered as Software as a Service (SaaS), usually as a subscription

We can expect a commercialised ML product to be faster, more stable and more refined than a demo on github, with a price tag to match.

Other ML Examples

Given the process outlined above - lets explore a few more ML ideas. Instead of starting on arxiv.org - lets jump to searching github instead, so we know the projects we find have a good chance of a functioning demo.

Searching Github

- real-time voice clone
- Text To Speech (TTS)
- Audio Source Separation (spleeter)

¹⁾

the ideepcolor training set is 1.3 million images

²⁾

this is obviously a contrived example of course, but the principle applies regardless

³⁾

if made available under an [appropriate license](#)