1) **PyTorch**

PyTorch has become one of the go-to frameworks for many researchers, because of its implementation of the novel Dynamic Computational Graph paradigm. When writing code using other frameworks like TensorFlow, CNTK or MXNet, one must first define something called a computational graph. This graph specifies all the operations that will be run by our code, which are later compiled and potentially optimized by the framework, in order to allow for it to be able to run even faster, and in parallel on a GPU. This paradigm is called Static Computational Graph, and is great since you can leverage all sorts of optimizations and the graph, once built, can potentially run in different devices (since execution is separate from building). However, in many tasks such as Natural Language Processing, the amount of "work" to do is often variable: you can resize images to a fixed resolution before feeding them to an algorithm, but cannot do the same with sentences which come in variable length. This is where PyTorch and dynamic graphs shine, by letting you use standard Python control instructions in your code, the graph will be defined when it is executed, giving you a lot of freedom which is essential for several tasks.

2) **Caffe2**

The original Caffe framework has been widely used for years, and known for unparalleled performance and battle-tested codebase. However, recent trends in DL made the framework stagnate in some directions. Caffe2 is the attempt to bring Caffe to the modern world.

It supports distributed training, deployment (even in mobile platforms), the newest CPUs and CUDA-capable hardware. While PyTorch may be better for research, Caffe2 is suitable for large scale deployments as seen on Facebook.

3) Pendulum

Last year, Arrow, a library that aims to make your life easier while working with datetimes in Python, made the list. This year, it is the turn of Pendulum.

One of Pendulum's strength points is that it is a drop-in replacement for Python's standard datetime class, so you can easily integrate it with your existing code, and leverage its functionalities only when you actually need them. The authors have put special care to ensure timezones are handled correctly, making every instance timezone-aware and UTC by default. You will also get an extended timedelta to make datetime arithmetic easier.

Unlike other existing libraries, it strives to have an API with predictable behavior, so you know what to expect. If you are doing any non trivial work involving datetimes, this will make you happier! Check out the docs for more.

4) Dash

You are doing data science, for which you use the excellent available tools in the Python ecosystem like Pandas and scikit-learn. You use Jupyter Notebooks for your workflow, which is great for you and your colleagues. But how do you share the work with people who do not know how to use those tools? How do you build an interface so people can easily play around with the data, visualizing it in the process? It used to be the case that you needed a dedicated frontend team, knowledgeable in Javascript, for building these GUIs. Not anymore.

Dash, announced this year, is an open source library for building web applications, especially those that make good use of data visualization, in pure Python. It is built on top of Flask, Plotly.js and React, and provides abstractions that free you from having to learn those frameworks and let you become productive quickly. The apps are rendered in the browser and will be responsive so they will be usable in mobile devices.

5) FlashText

When you need to search for some text and replace it for something else, as is standard in most data-cleaning work, you usually turn to regular expressions. They will get the job done, but sometimes it happens that the number of terms you need to search for is in the thousands, and then, reg exp can become painfully slow to use.

FlashText is a better alternative just for this purpose. In the author's initial benchmark, it improved the runtime of the entire operation by a huge margin: from 5 days to 15 minutes. The beauty of FlashText is that the runtime is the same no matter how many search terms you have, in contrast with regexp in which the runtime will increase almost linearly with the number of terms.

FlashText is a testimony to the importance of the design of algorithms and data structures, showing that, even for simple problems, better algorithms can easily outdo even the fastest CPUs running naive implementations.