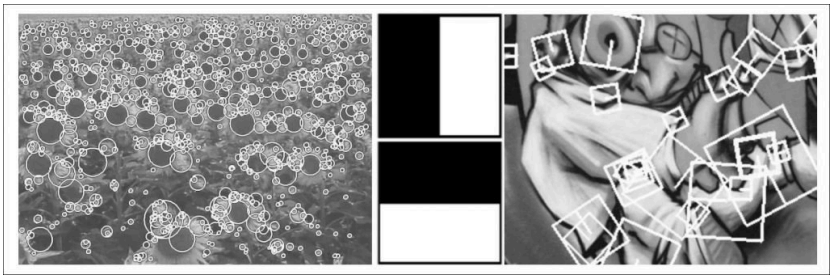
Features in Machine Learning

Originally published at https://blog.metaphysic.ai/features-in-machine-learning/

January 19, 2023

Since our central preoccupation is the neural reconstruction and manipulation of human faces/identities, the term 'feature' is unfortunately ambiguous, and has a far wider application in machine learning.
In literal terms, a feature is an input variable in the training data for any ML system. If you're training a spam detection system, for instance, the entities being studied during training could include variables such as *text content*, *email addresses*, *time-stamps*, and *specific phrases*.
If you're studying faces during training, features could represent the mean average pixel distance between recognized *facial* features (there's that ambiguity again) in a particular identity, such as the mapped distance relationship between the center of a person's two pupils to the recognized apex of the nose.
Therefore, in accordance with the central meaning of the word, a feature is a characteristic of a larger perceived entity. Despite the redundancy, features are also sometimes referred to as *feature variables*.



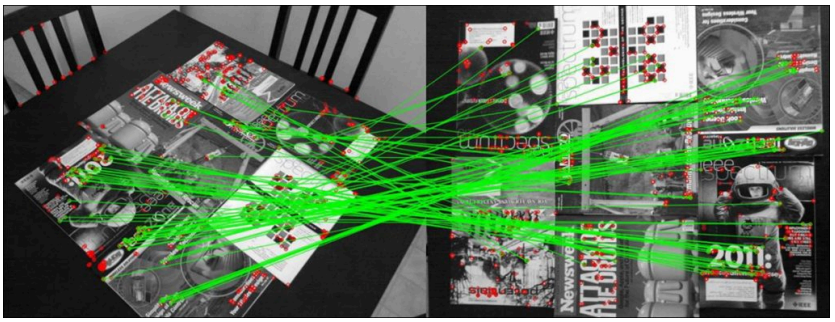
Detected interest points in a picture of a sunflower field, from the 2006 feature extractor SURF (see below). Source: https://people.ee.ethz.ch/~surf/eccv06.pdf

Types of Feature

A feature, in the sense we intend, will function as an input to a machine learning model. Features are identified (hopefully) from the training dataset during the training process, and are the central operating units during *inference* (the point at which the user can enjoy the functionality of the trained model).
There are many types, and sub-types of feature: a *nominal* feature describes an arbitrary name, category or label, such as '*woman'*, *'Tom Cruise'*, *'fruit'*, '*work'*, etc.
A *numeric* feature (or *quantitative variable*) is itself a host for two sub-types of numerical features: a *continuous* feature and a *discrete* feature.
A *continuous* feature is a range-bounded number, such as a person's age, or the distance between two cities. Since the current known upper limit for age is 122, and the maximum possible distance between cities (existing or yet to come) is 20,004 km (12,430 miles), these numbers have definite bounds, but can be represented in fractions and using any chosen scale (for instance, a person who is 2.1 meters tall can be registered as 6.80564e-17 tall, in parsecs).
A *discrete* feature is largely the same as a continuous feature, except that its possible values must be exact. The most famous example of this is the idea of having *2.4 children* – a demographic 'averaging' that results in a biological impossibility. Therefore discrete features are applicable for non-granular entities such as *household size*, or any other representation that cannot logically be expressed in fractions.
An *ordinal* feature is an entry in a range of features that can be sorted by rank, such as a grade-marking schema for student's exams.
There are other and varied taxonomies of feature types, some more specific to certain sectors. For instance, an *interval* feature cannot have a value of true zero; an example of this would be a bad weather score, or a nuclear alert rating system, where only positive values have any meaning. By contrast a *ratio* variable can have a value of zero or less, and would be applicable, for instance, to recording temperatures.

Feature Extraction for Images

Feature extraction for image content does not diverge from these base models, since all input data will end up in some way as text or numerical variables; however, the journey is a bit longer than for data that's nearer to this format already (such as tabular or general numeric data).
Machine learning architectures are essentially hyper-accelerated number crunchers. When the data is itself already numeric, little conversion is necessary to store it in a suitable and interpretable format during training, and not much conversion is needed when deriving features from the data either.
Other types of data, such as images and words, need to be converted into numbers in order to pass through the training process, with the same logic converting the transformed information back to an apposite state at inference time.
In the case of images, the base information that will be passed to the training system is already rationally laid out in a pixel array (i.e., the core pixels of a 64x64px or 512x512px image, for example). Even if the conversion and interpretation of images is a challenge, at least the medium is consistent.
One popular feature detector is Oriented FAST and rotated BRIEF (ORB), a successor to the 2004 SWIFT project, from the University of British Columbia, and SURF, from the Katholieke Universiteit Leuven in Belgium.

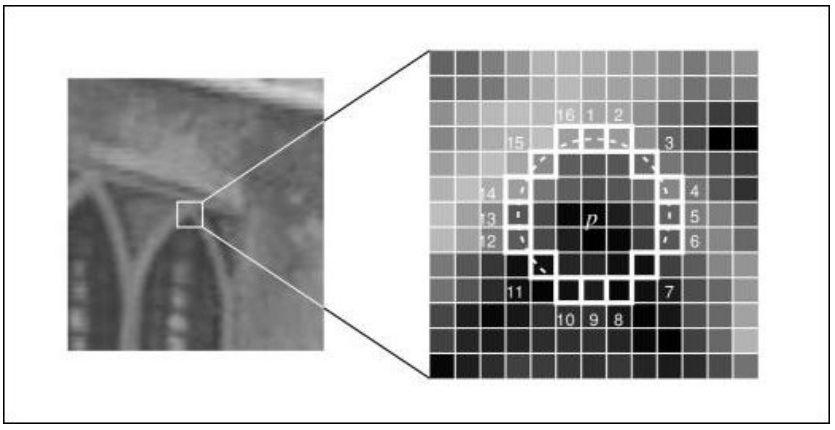


ORB at work, in its launch in 2011. The red points are unmatched, the green points are extracted and recognized features. Source: https://web.archive.org/web/20180107122611/http://www.vision.cs.chubu.ac.jp/CV-R/pdf/Rublee_iccv2011.pdf

ORB built on those prior works, as well as on BRIEF (Binary Robust Independent Elementary Features) from the CV Lab at the Swiss Federal Institute of Technology at Lausanne, and was among the first framework to offer real-time feature detection, whilst being (the paper asserted) two orders of magnitude faster than the contemporary state-of-the-art.

<YOUTUBE LINK OMITTED>

In 2017 one study found that though ORB was the fastest algorithm, SIFT performed best in most scenarios. The paper also noted that ORB concentrates more on regions in the center of the image, which indicates that it is using human-style framing as a proxy for possible regions of interest, while (the authors asserted) the keypoint detectors of SURF, SIFT and the FAST algorithm were distributed evenly over the image – a more laborious pixel traversal that's logically going to affect latency.



Feature detection with FAST. Source: https://docs.opencv.org/3.4/df/d0c/tutorial_py_fast.html

These technologies were preceded by many others, and research into extracting better 'regions of interest' is ongoing. Other approaches include Fast Retina Keypoint (FREAK) and Binary Robust Invariant Scalable Keypoints (BRISK).
Many of the image feature extractors mentioned employ Random sample consensus (RANSAC) methodology to make sense of the points identified, and to resolve them into central shapes, lines, and the delineations of images, such as faces. RANSAC resolves these kinds of groups by identifying outliers from the established points of interest; whatever is left behind after discounting outliers is established as some kind of central cohesive point of interest, such as the margin of an object.

<YOUTUBE LINK OMITTED>

It has been noted that the RANSAC approach is far from invariant or resistant to its environment, and needs fitting to the characteristics of different feature detectors.