Machine learning terminology

Tom Rochette <tom.rochette@coreteks.org>

March 22, 2020 — 57bd6871

0.1 Context

Machine learning is a complex field and as any other field, it has its share of terminology one needs to get acquainted with to speak with other practitioners more succinctly.

The following is my attempt at describing many of the terms I use on a regular basis. I don't claim many of them are perfect, I would even say that some are potentially wrong, which is the whole reason I am doing this exercise. It is a good way for me to discover holes in my understanding and to fix them.

If you think any of the descriptions below are wrong, feel free to let me know through the comments or a PR/issue on the github repository.

0.2 Learned in this study

0.3 Things to explore

1 Overview

Activation function Given a value, the activation **function** returns a value (they are functions after all!). The purpose of activation functions are to return a value based on the sum of all the input arriving at the unit. Certain activation functions will return -1 or 0 for very large negative values and 1 for very large positive values (sigmoid/logistic, tanh or softsign). Other types of activation functions will not saturate (e.g., min to -1 or max to 1), which may cause the network to become unstable due to numerical instability of operations of large numbers against small numbers (or small numbers with small numbers and large numbers with large numbers). #neural-network
Add-one smoothing Assign a count of 1 to unseen **n-grams**. This is done so that they at least have a very minor **probability** instead of none. #nlp
BLAS Basic Linear Algebra Subprograms. Generally, those are optimized **linear algebra procedures** that can be executed more efficiently on certain hardware.
CNN Convolutional neural network. A type of neural network **layer** based on the concept of applying **convolutions**. CNNs are useful for translation invariance, that is, they will detect **features** wherever they are located within the image. CNNs are mostly used in the context of vision/image problems or problems where the space structure provides additional information the network may benefit from. #cnn #neural-network
Convolution The concept of convolution can be seen as the operation of applying a **filter/kernel** on a **tensor**. A convolution is generally the operation of multiplying the **kernel** (another tensor with defined values) at each location in the tensor with the associated cells, then summing all the associated values, which becomes the value of the cell in the produced tensor. #cnn
Data A collection of values which can be serialized in binary format.
Decoder An abstraction of a **neural network** which takes a compressed **representation** and decompresses it into its original representation. #neural-network
Derivative #mathematics
Domain The set of values for which the **function** is defined. #mathematics
Dimension Represent the number of numbers one needs to represent a coordinate in a **space**. #mathematics

Embedding A **tensor** which purpose is to encode some information in a different space than the input space. For instance, we might be encoding a phrase as a **sequence** of unique integers. We could also transform these sequence of integers into sequence of **vectors** as a pre-processing step, using an embedding that was learned and which converts word indexes into vectors. Embeddings can sometimes be used to create more compact **representations** of their input, such as converting a large **one-hot vector** (rank/length 100<) into a small vector (rank/length 10 for example). #neural-network

Encoder An abstraction of a **neural network** that takes an input and compresses it into a more compact **representation**. #neural-network

Epsilon-greedy strategy/policy An action in a set of actions A is selected at random with **probability** ϵ while we may pick the action that gives the maximum amount of reward (greedy) with probability $1 - \epsilon$. #reinforcement-learning

Example

Feature A property of the **data** used to do **prediction**. An example of a feature would be the length of items, or their color, their price, etc. Features can have various types such as numeric, categorical (labels), discrete or more complex such as **vectors**, **matrices**, **tensors**.

Filter A **tensor** used in a **convolution** operation. #cnn

Fingerprinting Converting a large **space** into a small space for faster comparison. Common fingerprinting algorithms are hashes such as md5/sha1.

Function An **operation** which transform its input(s) (called arguments) into output(s). A function is expected to return the same outputs given the same inputs (as opposed to returning different outputs given the same inputs). A function is a useful concept because it allows one to abstract his thinking by hiding (or as it is called in computer science, encapsulating) it behind a function. In loose terms, a function represents a behavior/transition. #mathematics #computer-science

Function approximator A function approximator is a **function** that attempts to mimic another unknown function. A function approximator is useful because it may allow you to model an unknown function. In doing so, it allows you to rely on this approximation which may be faster to compute or require less resources than the function it mimics.

Gradient The gradient of a **function** is the "slope", or amount of change, that occurs at a given point of the function. The gradient is a **vector** composed of the n **partial derivatives** of f. #neural-network

Hash function A type of **function** that converts a large **space** into a small space, where it is difficult to generate the domain values given image values. Examples of hash functions are md5/sha1. #mathematics

i.i.d. All the **random variables** are independent (the occurrence of one does not affect the others) and identically distributed (they have the same **probability distribution**).

Image See **range**.

Kernel See **filter**.

Layer In a **neural network**, a layer is an ensemble of units that receive inputs and generate outputs. The outputs are generally computed through an **activation function**. Each unit is responsible of computing some **function** on its input, which is then provided to the activation function to determine the unit's output. Generally, the units of a layer can be computed in parallel as they do not depend on one another. ##neural-network

Linear algebra

Loss function A **function** that returns a **metric** which can be **optimized** (minimized or maximized). In the case of a loss function, we want to minimize this value. Examples of loss functions are the mean squared error or the categorical cross-entropy. A loss function is computed against two values, one being a training **example** (the target) and the other a **prediction** (the current state). #neural-network

Matrix A collection of numbers arranged in an array, generally represented to be within the $\mathbb{R}^{m \times n}$ **space**, where m and n represents the row size and column size respectively. A matrix is a special case of a **tensor**; its **tensor rank** is 2 and its **tensor shape** (m, n) represents its row and column **dimensions**. #mathematics

Maximum likelihood

Metric

n-gram A **sequence** of n items, generally from a sequence of text. n-grams are generally used as atomic units in other systems, such as **neural networks**, where instead of representing two words, the bigrams are given a unique **nominal number** and then are **one-hot encoded**. #nlp

Neural network A neural network is a **function approximator**. It is generally composed of **layers** which transform their inputs according to a **function**. Neural network are trained, in that given a **training data set** (inputs/outputs), we want the neural network to learn to generate the same outputs given the same inputs. Training occurs through the process of modifying the **weights** of each layer. #neural-network.

Nominal number A number (generally an integer) that serves as a unique identifier for a more complex value, for instance a string. A nominal number can be thought of as the primary key in a SQL table, it only serves to uniquely represent the row but generally has no meaning by itself. #mathematics

Normalization The process of converting numbers with arbitrary range so that they are contained within the [-1, 1] or [0, 1] range. This is mainly done because within **neural networks** the scale of various **features** can have an impact on the first **layer** and any subsequent layers if the first layer does not saturate.

Observation

One-hot encoding The process of converting a **nominal number** into a **One-hot vector**, that is to say convert a value that is generally considered as part of an enumeration into a different **representation**. The purpose of one-hot encoding is to convert something that isn't numeric into a **tensor** representation. The reason **nominal numbers** can't be used directly in a **neural network** is that they do not represent linearity in the **feature**, that is to say nominal numbers 1, 2, 3, 4 representing "dog", "cat", "horse", "bird" do not represent that the "intermediate" of a dog and a horse is a cat; every item is a unique entity. This process thus turns a single feature into multiple features that represent the presence or absence of this nominal value (yes, this is a bird, no it is not a dog, a horse or a cat). #neural-network

One-hot vector A **vector** which is 0 in most **dimensions** and 1 in a single dimension. Ex: [0, 0, 0, 0, 0, 1, 0, 0, 0]. One-hot vectors are generally used to encode **nominal number**. #neural-network

Operation See **function**.

Optimization The process of minimizing or maximizing a **function** (finding the coordinates at which the function returns its maximal or minimal value). #neural-network

Partial derivative #mathematics

Pooling layer Given a certain pool size (a **vector** or **tuple**), the pooling layer applies an operation on a "block" (sub-tensor) of data. For instance, max pooling will, given a vector such as [1 2 3 4] and a pool size of 2, return the vector [2 3 4], with a pool size 3, return the vector [3 4], and with a pool size 4, return [4]. Max pooling is often used in **CNNs** in order to select the highest value in a rectangular region of the image. The pooling operation is similar to a **convolution** in that it applies to a sub-region of the data and that the output is resized based on the kernel used. #cnn

Prediction Based on a set of **observations** (input and output) and given an input, returns a prediction as to what the output is expected to be.

Probability The likelihood that an event will occur. Between 0 and 1, where 0 implies it will not happen and 1 implies it will, and values in between indicating the likelihood it will occur. #mathematics

Probability distribution A mathematical **function** that describes the **probabilities** of occurrence of a given event in a set of events. #mathematics

Procedure A sequence of operations.

Random variable A **variable** which may take out one of many potential values. This random variable has a **probability distribution**, which defines the probability of each potential value it can take. #mathematics

Range The set of values a **function** takes on as output. #mathematics

Recurrence A recurrence is an which constructs the values of a **sequence** based on the previous elements of that sequence. #mathematics

Representation A binary encoding (which can in turn be numbers, characters, images, audio, etc.) of data.

RNN Recurrent **neural network**. A type of neural network **layer** based on the concept of **recurrence**. RNNs are particularly well suited for sequential data. RNNs expect **sequences** of fixed length n, which allows them to be unrolled into a "regular" network where the current state (at time t) is computed based on the previous state (at time $t-1$) and the current input (at time t). You can think of RNN as nested dense layers applied on the sum of the input and a tensor $\boldsymbol{U}$ and the previous state and a tensor $\boldsymbol{W}$, such that $s_t = f(\boldsymbol{U}x_t + \boldsymbol{W}s_{t-1})$, where f is an activation function. #rnn #neural-network

Sequence A list (ordered collection) of numbers. #mathematics

Skip-grams Generalization of **n-grams** in which the components (typically words) need not be consecutive in the text under consideration, but may leave gaps that are skipped over. Thus, in a sentence such as "This is a simple example sentence.", a skip-gram could be "This a", "is simple", "a example", "simple sentence" for a 1-skip bigram. In the same fashion as **n-grams**, probabilities would be associated with the skip-grams,

allowing our system to better predict subsequent items. #nlp
Space See **vector space**. #mathematics
Supervised learning The process of **training** using a **training data set**, where the inputs and outputs are both known and we are trying to construct a **function approximator** using this information.
Syntactic n-grams n-grams defined by paths in syntactic dependency or constituent trees rather than the linear structure of the text. Syntactic n-grams can be thought as **skip-grams** where the distance varies based on the length of the syntactic item. #nlp
Tensor A multi-**dimensional** array containing numbers. A tensor is represented by a **rank**, a **shape**, and a **type**.
Tensor rank The number of **dimensions** of the tensor. Examples are rank 0 for a scalar, rank 1 for a vector, rank 2 for an image/matrix, rank 3 for a video (**sequence** of images in time).
Tensor shape The number of elements within each **dimension**. For example, a tensor with shape [5] can contain [a, b, c, d, e] while a tensor with shape [1, 4, 3] will be [[[1, 2, 3], [1, 2, 3], [1, 2, 3], [1, 2, 3]]].
Tensor type The type of data contained within the tensor. Generally it will be numbers (integers, floats, double), but it can also be boolean, characters or strings.
Training The process of **optimizing** a set of weights in a **neural network** in order to improve the predictive ability of the network. Training is generally done in a **supervised** manner, using a **training data set**. #neural-network
Training data set A set of data that is used to **train** a **neural network** to recognize the data within the data set and to generate an expected response. #neural-network
Tuple An finite ordered collection of items of various types. As tuples are finite (have a fixed length), they are sometimes considered more "conceptually" appropriate than using lists or arrays which could increase in size/length. #mathematics
Universal approximation theorem
Variable A placeholder that can take a value out of many different possible values.
Vector A collection of numbers, generally represented to be within the $\mathbb{R}^n$ space, where n represents the length of the vector. Each value of the vector represents a value within a single **dimension** of the **vector space**. A vector is a special case of a tensor; its **tensor rank** is 1 and its **tensor shape** (n) represents its length. It is also a special case of a matrix, where the matrix has only 1 column. #mathematics
Vector space A vector space is an n-**dimensional** space constructed through the addition and multiplication of vectors. #mathematics
Weights A **tensor** that is **trained** (modified) in order to learn to approximate a given function. Weights are initialized either with a specific distribution, or randomly. During training, the values within the tensor are updated until they approximate the data the best (according to a **loss function**). #neural-network

2 See also

3 References

- https://developers.google.com/machine-learning/glossary/
- http://www.wildml.com/deep-learning-glossary/
- http://www-anw.cs.umass.edu/rlr/terms.html
- https://machinelearning.wtf/
- https://deeplearning4j.org/glossary
- https://yanndubs.github.io/machine-learning-glossary/
- https://www.analyticsvidhya.com/glossary-of-common-statistics-and-machine-learning-terms/