An Introduction to Convolutional Neural Network Benchmarks and Metrics

Landon Buell

27 January 2020

1 Introduction

Tom Mitchell, former Chair of Machine Learning Department at Carnegie Melon University, describes *machine learning* as:

> A computer program is said to learn from experience E with respect to some task T, and some performance measure P, id it's performance on T, as measured by P, improves with experience, E. [3]

As the complexity of machine learning algorithms increase, it's important that we can still measure the performance, P, as described by Mitchell. In terms of programs structures like Convolutional Neural Networks (CNNs), there exists various sets of benchmarks that we can use to evaluate the performance of that network. Each metric has a particular use and validity and each one may only apply to particular type of algorithm. In this introduction, I will break up mthese benchmarks into two very board categories: *Regression Metrics* and *Classification Metrics*.

For this work I will also use a notation convention as used by James Gareth in his book "An Introduction to Statistical Learning". A value or function that is approximated or estimated with set of data X, is denoted with a *hat*. Thus an approximated function, f becomes $\hat{f}$ which produces an estimated output $\hat{Y}$, which can be compared to known outputs, Y. Additionally, X and Y can be treated as discrete vector-like objects, which can be indexed through with a subscript. Thus sets X and Y, which n elements are equivalently:

$$X = [x_1, x_2, x_3, ..., x_i, ..., x_{n-1}, x_n] \tag{1}$$

and

$$Y = [y_1, y_2, y_3, ..., y_i, ..., y_{n-1}, y_n] \tag{2}$$

2 Regression Metrics

Generally, a regression algorithm is a type of Machine Learning Algorithm that seeks to build a function f, using a set of data X such that the result of the function is continuous, real number value [5]. In function notation:

$$f : X \to \mathbb{R} \tag{3}$$

A regression algorithm generally takes some function and applies a *best fit* line or curve through the data, X as a method of approximation [2]. The output of that function is $Y \in \mathbb{R}$.

The estimated function $\hat{f}$ operates as:

$$\hat{f}(X) = \hat{Y} \tag{4}$$

or for each individual element of X:

$$\hat{f}(x_i) = \hat{y}_i \tag{5}$$

2.1 Mean Squared Error

Mean squared error (MSE) is a very basic way of quantifying how well the predicted response matches a known set of responses. To compute this value, we take the known output for a sample, y_i and find it's difference from the predicted output $\hat{f}(x_i)$, and then the result is squared. This quantity is repeated for each index, i, and divided by the total number of elements in the set, n.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - \hat{f}(x_i)\Big)^2 \tag{6}$$

So if the predicted response from $\hat{f}$ differs very slightly from each index in the expected response, y_i, then the whole MSE takes on a very small values. This indicates a regression that fits the data well. Conversely, if the predicted response differs greatly from each index, the the whole MSE error takes on a very large value, and indicates that the data is not fit very well. In general, the goal of regression algorithms is to minimize the MSE value [5].

3 Classification Metrics

A classification algorithm is a type of Machine Learning Algorithm that seeks to build a function f, using a set of data X, such that the result of the function is one of k classes [5]. In function notion:

$$f : X \to \{1, 2, ..., k-1, k\} \tag{7}$$

A classification algorithm takes some set of data, X, and using qualities or *features* of that data, places each element, x_i into a bin, called a *class* with other similar elements [3].

3.1 K - Folds Cross Validation

K - Folds Cross Validation (also called K-Folds X-val) is a method that is part of a larger set of cross validation algorithms. K- Folds is a resampling procedure that allows for the evaluation of a certain model based on a limited set of data [1]. For the algorithm to work, it requires that there be a data set X, that is fully labeled, and can be broken down into an arbitray amount of subsets.

The procedure revolves around the parameter k, which gives the algorithm the amount of sub groups that sample data is split into for the duration of it's execution. Very generally, the procedure of the algorithm is as follows:

1. Shuffle data set X as needed. Split the data set into k similarly sized subsets. $X \rightarrow \left[X_1, X_2, ...X_k\right]$.
2. For each set of data, X_n, save it as a testing subset. Use the remaining $k-1$ subsets as a the set of training data for the model.
3. Fit the model with $k-1$ subsets and evaluate. Hold the evaluation (can be percentage accuracy) and discard the particular model.
4. Repeat steps 2 and 3 for the remaining $k-1$ subsets.

After the procedure has been completed for all k subsets, the results can be amalgamated as desired. This method ensures that every sample in the entire original data set, X is used at least once as a testing element, and $k-1$ times as a training element. The exact value of k is contingent on the particular data set. No single value is useful for all types of models. Often, k should be chosen such that each subset is also statistically similar to the larger data set [3]. Additionally, the value $k = 10$ has been experimentally shown to produce generally low-biased estimates on the performance of a classifier [1].

it is important to recognize that percent accuracy alone is a very poor way of determining the validity of model [3]. K- Folds alone provides very little valuable information, and thus in often used in conjunction with other benchmark methods.

3.2 Precision Score

3.3 Recall Score

3.4 F1 Score

3.5 Confusion Matrix

References

[1] Brownlee, Jason. “A Gentle Introduction to k-Fold Cross-Validation.” *Machine Learning Mastery*, 8 Aug. 2019, machinelearningmastery.com/k-fold-cross-validation/.

[2] James, Gareth, et al. An Introduction to Statistical Learning with Applications in R. Springer, 2017.

[3] Géron Aurélien. *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O’Reilly, 2017.

[4] Goodfellow, Ian, et al.*Deep Learning*. MIT Press, 2017.

[5] Petrik, Marek. “Introduction to Machine Learning.” Machine Learning. 22 Jan. 2020, Durham, New Hampshire.