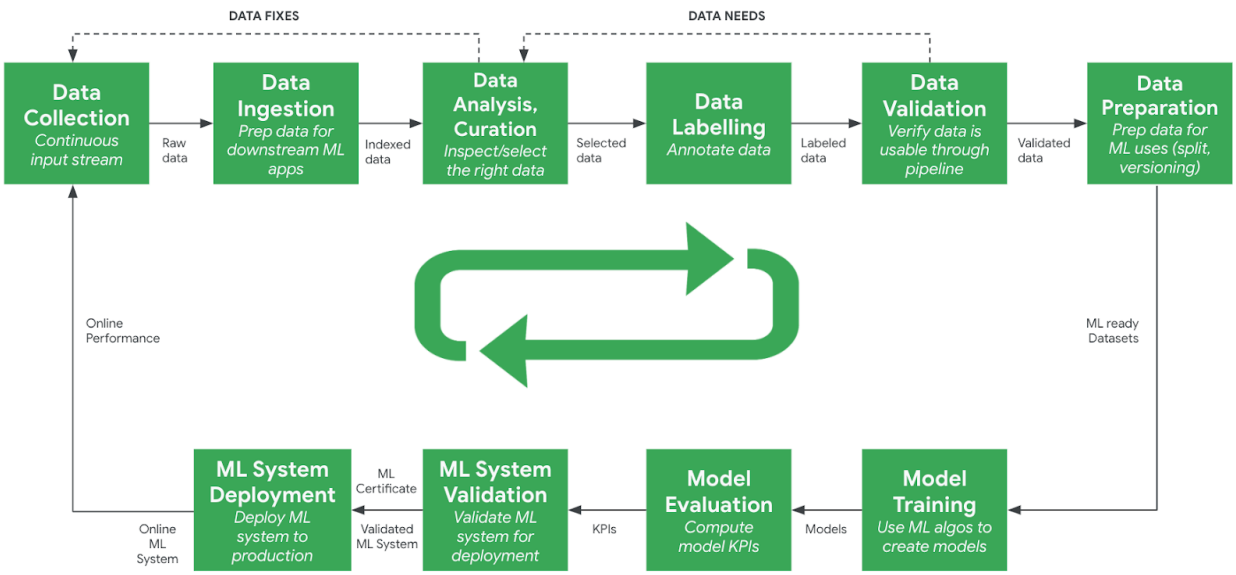
Machine Learning Lifecycle



The development of a machine learning model follows a multi-step design methodology, commonly referred to as the machine learning lifecycle. Diagrams typically illustrate this lifecycle using a varying set of discrete steps. This methodology is also applicable to tiny machine learning applications, albeit with different requirements to conventional machine learning models. Some of the most important steps in the machine learning lifecycle are as follows:

Design Requirements. The first stage of the machine learning lifecycle is where the application requirements are established. For tiny machine learning applications, hardware constraints (e.g., memory and storage footprint, latency) are often the most pressing requirements. Other requirements can include temporal resolution of data, the number of inferences per second, or minimum model accuracy. This step, whilst being the least well-defined, is the most crucial step, as it sets the scene for the remainder of the lifecycle. The more clearly defined the design criteria are, the easier it is to evaluate the model in later stages. Data requirements, such as the variable of interest and potential feature variables, can also be specified during this stage.

Data Collection/Curation. In this stage, design requirements are known and data begin to be collected that are suspected to aid in predicting the variable of interest. This can often involve active data collection or curation of information from external sources, such as pre-existing datasets (e.g., CIFAR10, ImageNet). This is often one of the most time consuming stages, as it can be difficult and time-consuming to curate a sufficiently large dataset, simultaneously ensuring to minimize dataset bias. No data analysis is

performed during this stage. In Course 2, we will learn the challenges associated with data collection for different types of sensors that serve as inputs for tiny machine learning applications.

Exploratory Data Analysis/Data Preprocessing. During this phase, the collected data is analyzed to determine which features are most informative at predicting the variable of interest. Feature engineering is common at this stage, extracting new information from available data. An example of this would be to extract the day of the week or whether it is a weekend from a time variable. Preprocessing involves ensuring the data follows standard modeling assumptions (e.g., are normally distributed) or manipulating data to meet these assumptions (e.g., log transformations on highly skewed variables). Data imputation is also typically done to appropriately fill in or remove missing data, and invalid or duplicate data is handled accordingly. When we develop our keyword spotting model, we will learn how to pre-process the audio signal.

Model Development. All of the previous stages focused on the design aspects, as well as the procurement and analysis of data. These stages often make up the dominant proportion of the time in the machine learning lifecycle. Once a viable dataset is available, the next stage is creating a suitable machine learning model. There are a plethora of machine learning techniques that have various pros and cons. One of the jobs of a machine learning practitioner is selecting an appropriate model for the task at hand. For example, basic linear regression may be suitable in environments where interpretability is key, but are constrained to the set of linear functions. This may present bias in situations where the distribution of data is highly non-linear in feature space, which can result in poor model accuracy. Conversely, neural networks may offer high performance as they are able to model the non-linear data distribution more effectively, but are less interpretable to the user. Keeping in mind the design requirements is important during this stage to know on what grounds to evaluate different models. One thing we will do in Course 2 is to understand the different model development approaches. Training a neural network from scratch is not always necessary.

Model Validation. It is commonplace to generate multiple models and to compare their performance on an unseen data set, known as the test set. The presence of a test set helps to ensure that the model has effectively modeled the distribution of data and has not overfit to points in the training set. The design criteria help to determine what metrics should be used in comparing the performance of various models. For example, if accuracy is the dominant metric, the model with the highest accuracy should be chosen. This stage can also be used to determine whether specific features improved or hindered model performance (e.g., by analyzing the relative importance of features or by successively removing variables and retraining the model, known as ablation study).

But there is more to model validation than just looking at the model metrics. We will discover how to think about metrics in a new light, specifically, from an application deployment perspective.

Model Deployment. The last stage of the lifecycle involves the deployment of the model in the production environment. For tiny machine learning applications, this is one of the most important stages. The model size must be reduced sufficiently to fit on the embedded device through various means such as model compression (i.e., model distillation), type conversion (e.g.., changing model weights from floats to integers), and lossless compression (e.g., Huffman encoding). The model must then be compiled into a format compatible with the end device. We will learn about “quantization” and use it to optimize the model size for performance and latency to make sure we are building sufficiently ‘tiny’ models. We will use TensorFlow’s quantization API, as well as understand what’s going on behind the scenes.

After all of these stages, continuous monitoring of the model in the production environment is typically necessary to ensure that it is working effectively. If, after all of these steps, the performance of the model is unsatisfactory in the production environment, the model must be updated. This can be done using new design requirements, data, or modeling techniques. This is the essence of the machine learning lifecycle.