When compiling a ML model, selecting the right **loss function**, **optimizer**, and **metrics** is critical to achieving good performance. Here's a basic overview of these components, when to use which, and details about optimizers.

---

1. Loss Functions

The **loss function** measures the error between the predicted output and the true target values. Choosing the right loss function depends on the type of task.

For Regression

- **Mean Squared Error (MSE):** Penalizes larger errors more heavily; best for normally distributed targets.
- **Mean Absolute Error (MAE):** Robust to outliers; gives equal weight to all errors.
- **Huber Loss:** Combines MSE and MAE; good for datasets with outliers.
- **Log-Cosh Loss:** Similar to Huber but differentiable everywhere; handles outliers better than MSE.

For Classification

- **Binary Cross-Entropy (Log Loss):** For binary classification tasks.
- **Categorical Cross-Entropy:** For multi-class classification when classes are one-hot encoded.
- **Sparse Categorical Cross-Entropy:** For multi-class classification with integer labels instead of one-hot encoding.
- **Hinge Loss:** Used for training Support Vector Machines (SVMs).

For Probabilistic Models

- **Kullback-Leibler (KL) Divergence:** Measures how one probability distribution differs from a reference distribution.

For Custom Tasks

- **CTC Loss (Connectionist Temporal Classification):** Used for sequence prediction problems like speech-to-text or handwriting recognition.
- **Wasserstein Loss:** Used in GANs to address mode collapse.

---

2. Optimizers

Optimizers are algorithms that update the weights of a model to minimize the loss function.

Basic Optimizers

1. **SGD (Stochastic Gradient Descent):**
    - Updates weights using a small batch of data.
    - Requires manual tuning of the learning rate.
    - Best for convex problems and small datasets.

2. **Mini-batch Gradient Descent:**
    - Combines efficiency of SGD and batch gradient descent.
    - Commonly used for large datasets.

Adaptive Optimizers

1. **Adam (Adaptive Moment Estimation):**
    - Combines the benefits of RMSProp and Momentum.
    - Works well in most scenarios, especially for non-stationary objectives or sparse gradients.
2. **RMSProp:**
    - Normalizes gradients by dividing by the root of the moving average of squared gradients.
    - Effective for RNNs and problems with non-stationary data.
3. **Adagrad:**
    - Adapts learning rates individually for parameters based on past gradients.
    - Effective for sparse data but may lead to excessively small learning rates.
4. **Adadelta:**
    - Extension of Adagrad to address diminishing learning rates.
    - Keeps updates within a limited window.
5. **Nadam (Nesterov-accelerated Adaptive Moment Estimation):**
    - Variant of Adam with Nesterov momentum.
    - Accelerates convergence in many deep learning tasks.
6. **SGD with Momentum:**
    - Helps escape local minima by using momentum to smooth updates.

When to Use Which Optimizer

- **Small Datasets or Convex Problems:** SGD, with or without momentum.
- **General Deep Learning Models:** Adam is a go-to optimizer.
- **Recurrent Neural Networks:** RMSProp or Adam.
- **Sparse Data or High Dimensional Data:** Adagrad or Adam.
- **GANs:** Often requires experimentation, but Adam or RMSProp are commonly used.

---

3. Metrics

Metrics evaluate the model's performance during training and testing. Unlike loss functions, they don't directly affect model training.

Regression Metrics

- **Mean Absolute Error (MAE):** Measures average absolute difference.
- **Mean Squared Error (MSE):** Penalizes larger errors more.
- **R^2 Score (Coefficient of Determination):** Indicates how well the predictions fit the data.

Classification Metrics

- **Accuracy:** Percentage of correct predictions; useful for balanced datasets.
- **Precision, Recall, F1-Score:** For imbalanced datasets or when false positives/negatives have different costs.
- **AUC-ROC (Area Under Curve):** Evaluates the trade-off between sensitivity and specificity.
- **Logarithmic Loss:** Measures the accuracy of probabilistic predictions.

Specialized Metrics

- **BLEU/ROUGE Scores:** For natural language tasks.
- **IoU (Intersection over Union):** For object detection and segmentation tasks.
- **Perplexity:** For language modeling and sequence generation.

Best Practices for Choosing Loss, Optimizer, and Metrics

1. **Start Simple:**
    - Use standard combinations like `Adam + Binary Cross-Entropy` for binary classification.
    - For regression, try `Adam + MSE` or `SGD + MAE`.
2. **Task-Specific Tuning:**
    - For imbalanced classification, use weighted cross-entropy or focal loss.
    - For structured prediction, explore task-specific metrics like IoU or BLEU.
3. **Experimentation:**
    - Experiment with different optimizers (e.g., Adam, RMSProp) and learning rates.
    - Use callbacks like **learning rate schedulers** to dynamically adjust learning rates.
4. **Monitor and Adjust:**
    - Track multiple metrics during training.
    - Regularly evaluate on a validation set to avoid overfitting.

By understanding the nature of data and task, one can choose the optimal combination of loss functions, optimizers, and metrics for effective model training and evaluation.