INF721 - Deep Learning
L7: Evaluating Neural Networks

Prof. Lucas N. Ferreira
Universidade Federal de Viçosa

2024/2

1 Introduction

This lecture focuses on evaluating neural networks and other machine learning models. We will discuss how to properly split datasets, various evaluation metrics for regression and classification tasks, and how to interpret these metrics. Understanding how to evaluate models is crucial for selecting the best model and assessing its real-world performance.

2 Dataset Splitting

When evaluating machine learning models, it's important to assess their performance on unseen data. To accomplish this, we typically divide our dataset into three disjoint subsets:

- Training Set: Used to train the model
- Validation Set: Used to tune hyperparameters and select the best model
- Test Set: Used for final evaluation of the chosen model

2.1 Proportion of Dataset Splits

The proportion of data allocated to each subset can vary depending on the size of the dataset and the specific problem:

- Traditional Machine Learning (Low data regime, for example, 1K examples):
  - Train/Test: 70%/30%
  - Train/Validation/Test: 60%/20%/20%
- Modern Deep Learning (Big data regime, for example, 1M examples):

- Train/Test: 95%/5%
- Train/Validation/Test: 98%/1%/1%

It's important to note that in low data regimes, it's common to not have a separate validation set. In this case, the test set serves as both validation and test set.

2.2 Splitting Techniques

When splitting the dataset, it's crucial to simulate real-world scenarios:

- Uniformly at random: Suitable for i.i.d (independent and identically distributed) data, e.g., image classification tasks
- By time: Appropriate for data with temporal components, e.g., spam filtering or time series forecasting
- Never split alphabetically or by feature values

2.3 Cross-Validation

In low data regimes, using a single train-test split can lead to highly variable performance estimates. Cross-validation can help address this issue:

2.3.1 k-fold Cross-Validation

1. Split the dataset into k equal parts (folds)
2. For each fold i from 1 to k:
   (a) Use fold i as the test set
   (b) Use the remaining $k-1$ folds as the training set
   (c) Train the model and evaluate on the test set
3. Average the evaluation scores from all k iterations

2.3.2 Leave-One-Out Cross-Validation

This is a special case of k-fold cross-validation where $k = N$ (number of samples in the dataset). Each sample serves as a test set once, while the rest of the data is used for training.

2.4 Examples of Dataset Splits

Here are some examples of popular deep learning dataset splits:

- ImageNet (images):
  - 1.4 million images of 1000 classes
  - Train/Valid/Test: 90%/3%/7%
- MNIST (images):
  - 70K images of handwritten digits (10 classes)
  - Train/Test: 85%/15%
- Penn Treebank (sentences):
  - 46K sentences from Wall Street Journal
  - Train/Valid/Test: 85%/7.5%/7.5%
- MAESTRO Dataset (audio/MIDI):
  - 1276 classical music pieces
  - Train/Valid/Test: 75%/10%/15%

3 Imbalanced Datasets

In classification tasks, it's ideal to have a balanced distribution of classes. However, real-world datasets often have imbalanced class distributions, which can lead to several issues:

- Random splitting can produce splits with different class distributions
- Models might overfit to the majority class

3.1 Balancing Techniques

To address imbalanced datasets, several techniques can be employed:

3.1.1 Oversampling

Increase the number of minority class samples by:

- Duplicating existing samples
- Generating synthetic samples

3.1.2 Downsampling

Decrease the number of majority class samples by:

- Randomly selecting majority class examples to remove

3.1.3 Class Weighting

Assign weights to classes in the loss function:

$$L(h) = -\frac{1}{m}\sum_{i=1}^{m}[w_1 y_i \log(\hat{y}^{(i)}) + w_0(1-y^{(i)})\log(1-\hat{y}^{(i)})] \tag{1}$$

Where:

$$w_1 = \frac{n_0 + n_1}{2n_1}$$
$$w_0 = \frac{n_0 + n_1}{2n_0}$$

Here, n_0 and n_1 are the number of samples in the negative and positive classes, respectively.

4 Regression Evaluation Metrics

For regression tasks, most evaluation metrics are based on the residuals, which are the differences between the true values and the predicted values:

$$\text{Residual} = y - \hat{y} \tag{2}$$

4.1 Mean Squared Error (MSE)

$$MSE = \frac{1}{m}\sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2 \tag{3}$$

- Sensitive to outliers due to squaring
- Units: Squared units of the target variable
- Use when large errors are particularly undesirable (e.g., predicting stock prices)

4.2 Mean Absolute Error (MAE)

$$MAE = \frac{1}{m}\sum_{i=1}^{n}|y^{(i)} - \hat{y}^{(i)}| \tag{4}$$

- Less sensitive to outliers than MSE
- Units: Same as the target variable (easier to interpret than MSE)
- Use when you want to treat all errors equally (e.g., forecasting daily temperature)

4.3 Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2} \tag{5}$$

- Sensitive to outliers
- Units: Same as the target variable (easier to interpret than MSE)
- Use when you want a balance between MSE and MAE properties (e.g., estimating house prices)

4.4 Coefficient of Determination (R-squared)

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^{m}(y^{(i)} - \bar{y}^{(i)})^2} \tag{6}$$

- Measures the proportion of variance explained by the model
- Values range from 0 to 1 (higher is better)
- Scale-independent, allowing comparisons across different datasets

5 Classification Evaluation Metrics

For classification tasks, most evaluation metrics are based on the confusion matrix, which shows the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

5.1 Confusion Matrix

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Table 1: Confusion Matrix

5.2 Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

Proportion of all instances correctly classified.

5.3 Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

When the model predicts positive, how often it is correct. Use when false positives have a high cost. For example, in these applications, false positives are more costly or undesirable:

- Spam Detection: High precision ensures that legitimate emails are not incorrectly classified as spam. False positives (marking a legitimate email as spam) can lead to missed important communications.
- Content Recommendation Systems: In streaming services or e-commerce platforms, high precision means recommended items are more likely to be relevant to the user, improving user experience and engagement.
- Medical Diagnosis (for non-critical conditions): When screening for non-life-threatening conditions, high precision reduces unnecessary anxiety and follow-up tests for healthy individuals.

5.4 Recall

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

Proportion of actual positive instances that were correctly identified. Use when false negatives have a high cost. For example, in these applications, false negatives are more costly or dangerous:

- Cancer Detection: High recall ensures that as many cancer cases as possible are detected, even at the cost of some false positives. Missing a cancer diagnosis (false negative) can be life-threatening.
- Fraud Detection: In financial services, high recall helps catch as many fraudulent transactions as possible. The cost of investigating a false positive is often less than the potential loss from a missed fraud case.
- Autonomous Vehicle Obstacle Detection: High recall is critical to detect all potential obstacles, even if it sometimes leads to unnecessary braking. Missing an obstacle (false negative) could result in accidents.

5.5 F1 Score

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

Harmonic mean of precision and recall, providing a balanced measure. In many real-world scenarios, a balance between precision and recall is necessary:

- Search Engines: Need to return relevant results (high precision) while not missing important information (high recall).

- Product Quality Control: Must catch defective products (high recall) without unnecessarily rejecting good ones (high precision).
- Social Media Content Moderation: Should remove inappropriate content (high recall) without censoring acceptable posts (high precision).

6 Multiclass Classification

In multiclass classification problems, we extend the binary classification metrics to handle multiple classes. Let's consider a scenario with K classes.

6.1 Confusion Matrix

For a multiclass problem, the confusion matrix is a K × K matrix where the rows represent the actual classes and the columns represent the predicted classes.

	Predicted 1	Predicted 2	...	Predicted K
Actual 1	n_{11}	n_{12}	...	n_{1K}
Actual 2	n_{21}	n_{22}	...	n_{2K}
...	...	...	...	...
Actual K	n_{K1}	n_{K2}	...	n_{KK}

Table 2: Multiclass Confusion Matrix

Where n_{ij} represents the number of instances of class i that were predicted as class j.

6.2 Accuracy

Accuracy in multiclass classification is straightforward:

$$\text{Accuracy} = \frac{\sum_{i=1}^{K} n_{ii}}{\sum_{i=1}^{K} \sum_{j=1}^{K} n_{ij}} \tag{11}$$

This is the sum of correctly classified instances (diagonal elements) divided by the total number of instances.

6.3 Precision, Recall, and F1 Score

For multiclass scenarios, we calculate precision, recall, and F1 score for each class individually and then average them. There are two main approaches:

1. Macro-averaging: Calculate the metric for each class and then take the unweighted mean.
2. Micro-averaging: Calculate the metric using the sum of all true positives, false positives, and false negatives across all classes.

Here, we'll focus on macro-averaging as it gives equal weight to each class, which is particularly useful for imbalanced datasets.

- **Precision:**

  For each class i:

$$\text{Precision}_i = \frac{n_{ii}}{\sum_{j=1}^{K} n_{ji}} \tag{12}$$

  Macro-averaged precision:

$$\text{Precision}_{\text{macro}} = \frac{1}{K} \sum_{i=1}^{K} \text{Precision}_i \tag{13}$$

- **Recall:**

  For each class i:

$$\text{Recall}_i = \frac{n_{ii}}{\sum_{j=1}^{K} n_{ij}} \tag{14}$$

  Macro-averaged recall:

$$\text{Recall}_{\text{macro}} = \frac{1}{K} \sum_{i=1}^{K} \text{Recall}_i \tag{15}$$

- **F1 Score:**

  The F1 score for each class i is the harmonic mean of precision and recall for that class:

$$\text{F1}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \tag{16}$$

  Macro-averaged F1 score:

$$\text{F1}_{\text{macro}} = \frac{1}{K} \sum_{i=1}^{K} \text{F1}_i \tag{17}$$

6.4 Interpretation

In multiclass scenarios:

- Accuracy provides an overall measure of correct classifications but can be misleading for imbalanced datasets.
- Macro-averaged metrics give equal weight to each class, which is useful when all classes are equally important, regardless of their frequency in the dataset.

- Micro-averaged metrics (not shown in formulas) give more weight to classes with more instances and can be useful when class imbalance reflects the natural distribution you want to capture.

When reporting results for multiclass problems, it's often helpful to provide both the confusion matrix and the macro-averaged metrics to give a comprehensive view of the model's performance across all classes.

7 Conclusion

Evaluating machine learning models is crucial for understanding their performance and selecting the best model for a given task. By properly splitting datasets, choosing appropriate evaluation metrics, and interpreting the results, we can make informed decisions about model selection and improvement. Remember that the choice of evaluation metric often depends on the specific problem and the costs associated with different types of errors.

Exercises

1. In a modern deep learning scenario with a large dataset (around 1 million examples), what is the recommended split for train/validation/test sets?

   (a) 60%/20%/20%
   (b) 70%/15%/15%
   (c) 80%/10%/10%
   (d) 98%/1%/1%

2. Which of the following is NOT a recommended technique for handling imbalanced datasets?

   (a) Oversampling the minority class
   (b) Downsampling the majority class
   (c) Class weighting in the loss function
   (d) Splitting the dataset alphabetically by class names

3. Consider three regression models evaluated on the same test set:

   - Model A: MSE = 25, MAE = 4, R-squared = 0.85
   - Model B: MSE = 30, MAE = 3.5, R-squared = 0.82
   - Model C: MSE = 20, MAE = 4.5, R-squared = 0.88

   If the task is to predict house prices and we want to minimize large errors, which model should be chosen?

   (a) Model A
   (b) Model B
   (c) Model C
   (d) Not enough information to decide

4. In a binary classification problem, if a model has 150 true positives, 40 false positives, 10 false negatives, and 200 true negatives, what is the model's precision? (Round to 2 decimal places)

   (a) 0.79
   (b) 0.81
   (c) 0.88
   (d) 0.94

5. Which evaluation metric is most appropriate when you want to balance between precision and recall in a classification task?

   (a) Accuracy
   (b) R-squared
   (c) F1 Score
   (d) Mean Squared Error