[DeepLearning.AI] Machine Learning Modeling Pipelines in Production

Notes by **José Hilario**

AutoML

- *Data Ingestion → Data Validation → Feature Engineering → Train Model → Validate Model.*
- **Neural Architecture Search (NAS).** Automate the design of NNs, outperforming handcrafted designs.
- NAS is at the core of AutoML. It consists of three parts: *Search Space → Search Strategy → Neural Architecture → Performance Estimation → Search Strategy → Repeat.*
- Types of search spaces: 1) *Macro:* NN from layers and connection types; 2) *Micro:* NN from cells (smaller networks).
- Several search strategies: 1) Grid search; 2) Random search; 3) Bayesian optimization; 4) Evolutionary algorithms; 5) Reinforcement learning.
- Measuring AutoML efficacy:
  - **Validation performance.** Prohibitive for large search spaces and complex networks.
  - **Lower fidelity estimates.** Relax the problem (e.g. subset of data; lower resolution; fewer filters per layer). Greatly reduces cost but underestimates performance. Relative ranking of architectures is not preserved through relaxations.
  - **Learning curve extrapolation.** Predicts the learning curve: 1) Extrapolates based on initial learning; 2) Removes poor performances; 3) Have surrogate model and use it to estimate the performance based on architectural properties.
  - **Weight inheritance / Network morphisms.** New architectures initial weights are based on previously trained networks. Network morphism modifies the architecture without changing the underlying function, inheriting knowledge from the parent network.
- AutoML on the cloud: 1) Amazon SageMaker Autopilot; 2) Microsoft Azure AutoML; 3) Google Cloud AutoML.

Dimensionality Reduction

- **Before.** 1) Domain experts selected features; 2) Designed feature transforms; 3) Small number of more relevant features were enough.
- **Now.** 1) Data generation and storage is less of a problem; 2) Squeeze out the best from the data; 3) More high-dimensional data having more features.
- Though NNs perform a kind of auto. feature selection, this is not as efficient as a well-designed dataset and model. Unwanted features can introduce unwanted noise in the learning.
- Curse of dimensionality.
  - Many common ML tasks like segmentation and clustering rely on computing distances between observations.
  - Problems with having high-dimensional spaces: 1) More features; 2) Risk of overfitting; 3) Distances grow more similarly (every observation appears equidistant from the rest); 4) Concentration phenomenon for Euclidean distance; 5) Usual metrics become less informative; 6) Less understandable for humans; 7) Need more resources (processing power, data) to make meaningful models.
  - Why are more features bad? 1) Redundancy and/or irrelevancy; 2) Hard to interpret and visualize; 3) More noise added than signal; 4) Hard to store and process data; 5) Overfitting; 6) Runtime and system memory requirements; 7) Solutions take longer to reach global optima.
  - How many features are required? This depends on: 1) Amount of training data available; 2) Complexity of decision surface; 3) Variance on training data; 4) Type of classifier used; 5) Predictive information available in features.
  - Feature engineering: 1) Come up with ideas to construct better features; 2) Select the right features to maximize predictiveness.
- Dimensionality reduction methods.
  1. **Algorithmic dimensionality reduction.** Select the best k-dimensional subspace for projection.
     - For classification, max. separation among classes. E.g., Linear Discriminant Analysis (LDA).
     - For regression, max. correlation between projected data and response var. E.g., Partial Least Squares (PLS).
     - For unsupervised, retain as much data variance as possible. E.g., Principal Components Analysis (PCA).
  2. **Principal Components Analysis (PCA).** Finds the principal components (PCs) of the data, to account for the variance of data in as few dimensions as possible using linear projections.
  3. **Singular Value Decomposition (SVD).** Decomposes non-square matrices, removing redundant features from the dataset. Most useful for sparse matrices.
  4. **Independent Components Analysis (ICA).** PCA seeks directions in feature space that minimize the reconstruction error; ICA seeks directions that are the most statistically independent. ICA addresses higher-order dependence.
  5. **Non-negative Matrix Factorization (NMF).** NMF models are interpretable and easy to understand. NMF requires sample features to be non-negative.

Quantization and Pruning

- Demand moves ML capabilities from cloud to on-device. Cost-effectiveness is a factor producing this shift, as well as compliance with privacy regulations.
- **Quantization.** Transform a model into an equivalent representation that uses parameters and computations at a lower precision. This improves the model's execution performance and efficiency, but it can often result in lower model accuracy. Concrete benefits are: 1) Faster computations; 2) Lower memory bandwidth; 3) Lower power consumption.
- The quantization process: 1) Quantization squeezes a small range of floating point values into a fixed number of information buckets; 2) This process is lossy in nature, but the weights and activations of a particular layer tend to lie in a small range, which can be estimated beforehand; 3) Affects static values (params), dynamic values (activations), computation (transformations).
- Strategies of quantization: 1) Post-training quantization: Joint optimization for model and latency, reduced precision representation; 2) Quantization-aware Training (QAT): Simulates low precision inference time computation in the forward pass of the training process (must avoid quantizing critical layers such as attention mechanisms).
- **Pruning.** Remove parts of the model that don't contribute to the performance. Model sparsity is the goal. Some benefits are: 1) Better compression for storage or transmission; 2) Gain speedups in CPU and some ML accelerators; 3) Combine with quantization.

High Performance Modeling

- Types of distributed training.
  1. **Data Parallelism.** Models are replicated onto different accelerators, and the data is split between them.
     - The frequency of parameter synchronization affects both statistical and hardware efficiency.
     - Synchronizing at the end of every mini-batch reduces the staleness of weights used to compute gradients, assuring good statistical efficiency. However, this requires each GPU to wait for the gradients from other GPUs. Communication can often dominate total execution time.
     - The factors that contribute to the increase of communication overhead across all models, are: 1) An increase in the number of data-parallel workers; 2) An increase in GPU compute capacity.
     - Challenges keeping the accelerators busy: 1) Limited memory; 2) Sequential nature of NNs leads to under utilization of accelerators; 3) Data parallelism can't

help increase the maximum model size supported by an accelerator.

2. **Model Parallelism.** When models are too large to fit in a single device, then they can be divided into partitions, assigning different partitions to different accelerators.

- Distributed training using data parallelism.
  1. **Synchronous training.** All workers train and complete updates in synchronized manner (all-reduce architecture).
  2. **Asynchronous training.** Each worker trains and completes updates separately (parameter server architecture). This method is more efficient but can result in lower accuracy and slower convergence.
- **High performance modeling.** Use input pipelines to keep the accelerators busy all the time. To optimize the pipeline performance: 1) Pre-fetch; 2) Parallelize data extraction and transformation; 3) Cache; 4) Reduce memory; 5) Order operations correctly.
- **Overcoming memory constraints:** 1) Gradient accumulation: Split batches into mini-batches and only perform back-propagation after the whole batch; 2) Memory swap: Copy activations between CPU and GPU accelerator, back and forth.
- **Pipeline parallelism (GPipe, PipeDream).** Integrates data and model parallelism. Divides mini-batches into micro-batches. Different workers for different micro-batches, in parallel. Allows ML models to have significantly more parameters.

Knowledge Distillation

- Concentrate the complexity of more sophisticated models into smaller networks, expressing the learning more efficiently, while duplicating the performance. The main idea is to create a simple *student* model that learns from a complex *teacher* model. We can think about the *student* model as a simplified—or compressed— version of the *teacher* model.
  - *Teacher* (normal training) maximizes the actual metric.
  - *Student* (knowledge transfer) matches the p-dist. of the teacher's predictions to form soft targets.
  - Train with joined objectives (*student* and *teacher*).

Model Performance Analysis

- Look at the model performance not just on the entire dataset, but on individual slices of the data. Top-level metrics may hide problems that occur in particular slices of the data.
- **Black-box evaluation.** Models can be tested for metrics like accuracy and losses like test error, without knowing their internal details.
- **Model introspection.** For finer evaluation: 1) Inspect which layers of the model learn to detect particular features; 2) Inspect the class activation map to determine which parts of the image are primarily responsible for influencing the model to produce the prediction of a particular class.
- **Model Robustness.** A model is considered to be robust if it is consistently accurate, even if one or more of the features change fairly drastically.
- **Model Debugging.** 1) Improve transparency of ML models by highlighting how data flows inside; 2) Improve fairness; 3) Reduce vulnerability to attacks; 4) Improve privacy; 5) Tackle model decay. Strategies: 1) Comparing to benchmark/baseline models; 2) Sensitivity analysis: Examine the impact of each feature on model prediction; 3) Residual analysis: Measures the difference between model predictions and ground truth (OK if uncorrelated).
- **Model Remediation.**
  1. *Data augmentation.* Add synthetic data into training set to help correct for unbalanced data.
  2. *Interpretable and explainable ML.* Overcome the myth of NNs as a black box. Understand data transformations.
  3. *Model editing.* Easily interpretable models allow for manual adjustments (e.g., decision trees).
  4. *Model assertions.* Apply business rules or simple sanity checks to the model, which override model predictions.
  5. *Discrimination remediation.* Increase diversity in the data; use fairness metrics to select hyperparameters and decision thresholds; learn fair representations.
  6. *Model monitoring.* Track accuracy, fairness, security problems, etc.
  7. *Anomaly detection.* Anomalies can be a warning of an attack. Enforce data integrity constraints.
- **Fairness.** Consider: 1) Context and different user types; 2) Seek domain-experts help; 3) Slice widely and wisely.
  - Conduct fairness tests on all the available slices of the data. A significant difference in a metric between two subgroups is a sign that the model may have fairness issues.
  - Evaluate fairness metrics across multiple thresholds to understand how they can affect the performance on different groups.
  - Inspect predictions which don't have a good margin of separation from their decision boundaries.
  - Measures.
    1. *PR/NR.* Appropriate when having equal final percentages of groups is desired.
    2. *TPR/FNR.* Measures equality of opportunity. Appropriate when it is important that the same percent of qualified candidates are rated positive in each group.
    3. *TNR/FPR.* Same as (2) but for the negative class.
    4. *Acc/AUC.* Appropriate when precision is critical.

Continuous Evaluation and Monitoring

- **Why to monitor?** The training data is a snapshot of the world at a point in time, while the real data continues evolving. Thus, trained ML models decay with time as well.
- **Statistical process control.** Drift detection (supervised). The number of errors is a binomial random variable; the alert rule is threshold-based.
- **Sequential analysis.** Linear four rates (supervised). If data is stationary, the contingency table should remain constant (precision, recall, specificity...).
- **Error distribution monitoring.** Adaptive Windowing (supervised). Calculate mean error rate at every data window. Size of window adapts (shorter when data is non-stationary).
- **Novelty detection.** Assign data to known cluster or detect emerging concept (unsupervised). Algs. available (OLINDA, GC3, etc.) Curse of dimensionality. Does not detect population-level changes. Helps detect emerging concepts.
- **Feature distribution monitoring.** Monitor each feature of the dataset separately (unsupervised). Split the incoming dataset into uniformly-sized windows, then compare the individual features against each data window. Algs. available (Pearson correlation, Hellinger distance). Not able to detect population drift since it only looks at individual features.
- **Model-dependent monitoring.** Concentrate efforts in monitoring the space near the decision margin in latent space (unsupervised). Margin Density Drift Detection (MD3). Areas in latent space where classifiers have low confidence are more important. A change in the margin density indicates drift. Helps to reduce the false alarm rate effectively.

Responsible AI

- **Fairness.** Ensure working towards systems that are fair and inclusive to all users.
- **Explainability.** Understand how and why ML models make certain predictions. Benefits: 1) Prevent wildly wrong results; 2) Defend against attacks; 3) Fairness; 4) Reputation and branding; 5) Answer to stake holders or customers who may question our model's decisions; 6) Legal or regulatory concerns.
- Models are interpretable if their operations can be understood by a human either through introspection or through a produced explanation.
  - The level of effort required needs to be feasible. One measure of interpretability of models is the amount of effort or analysis required to understand a given result.
  - Identify and validate relevant variables driving the model's outputs in order to develop trust in the reliability of the predictive system, even in unforeseen circumstances.
  - Understand: 1) Model-provided information useful to avoid prediction errors; 2) Which features drive the predictions.
  - Categories of model interpretation methods: 1) Intrinsic/ Post-hoc; 2) Model Specific/Agnostic; 3) Local/Global.
  - Results produced by interpretation methods are: 1) Feature summary statistics; 2) Feature summary visualizations; 3) Model internals; 4) Data point comparisons.
- **Privacy.** Training models using sensitive data needs privacy preserving safeguards.
- **Security.** Identifying potential threats can help keep AI systems safe and secure.