Latency-Aware Neural Architecture Search (NAS) with Multi-Objective Bayesian Optimization

**Problem**: Deploying models on end user devices such as mobile phones requires low-latency and accurate predictive models.

**Goal**: Provide an automated framework for identifying neural architectures that are optimal with respect to accuracy and on-device prediction latency.

**NAS Methodology**
- Pre-defined search space (i.e., # of layers, kernel shape)
- Search engine with a user-specified search strategy (i.e., multi-objective Bayesian Optimization)
- Selected architecture is trained/evaluated and deployed to obtain actual inference latency
- Inference latency is benchmarked on a tier-1 Android device with arm64 kernel through PyTorch lite interpreter
- Inference latency is defined as the time when the input data becomes available to the time the model generates the final outputs
- Accuracy/latency results are fed back to the search engine
- Depending on the objective, the search engine then selects the next candidate for evaluation
- Process is repeated until either the maximum number of iteration is reached or the final objective converges

**Use Case: On-Device Natural Language Understanding (NLU)**
- NLU is commonly used by conversational agents in most mobile devices and smart speakers
- Primarily objective:
  - Understand the user’s semantic expression
  - Convert the expression to a structured decoupled span form representation that can be understood by downstream task

**Model Architecture**
- Non-Autoregressive Transformer-like architecture
- Encoder generates the contextual information
- CNN-based length predictor predicts the length of the final output tokens
- Decoder to generate the info output tokens in one-shot

**Search Space**
- Search space includes kernel size, number/width of layers, number of attention head, etc
- A total of 24 parameters to be searched

**References**