**2.2 Neural architecture search**

The NAS has been introduced to automate the neural network design process [[6](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR6),[7](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR7),[8](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR8)]. NAS methods can be classified according to the *search mechanism* which is usually based on:

reinforcement learning [[8](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR8)] or

evolutionary algorithms (EA) [[32](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR32)];

however, many other search methods have been utilized [[7](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR7)]. During the search process, a common approach is to train and validate each candidate CNN to determine its accuracy. However, this is computationally expensive, even for data sets of moderate size. Hence, various alternative approaches such as surrogate models, one-shot search, and differentiable NAS have been proposed to accelerate the NAS method [[6](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR6), [7](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR7)].

NAS methods were initially constructed as single-objective methods to minimize the classification error [[33](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR33), [34](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR34)]. By introducing the hardware-aware NAS, the search process becomes multi-objective, and other objectives such as latency for a mobile phone or power consumption for an IoT node can be optimized. The best-performing CNNs obtained by NAS currently show superior quality concerning human-designed CNNs if the multi-objective scenario is considered [[35](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR35)].

As our NAS method employs genetic programming, we briefly discuss the main components of the EA-based approaches. Regarding the *problem representation*, direct [[35](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR35), [10](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR10), [34](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR34)] and indirect (generative) [[11](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR11)] encoding schemes have been investigated. The selection of *genetic operators* is tightly coupled with the chosen problem representation. While mutation is the key operator for CGP [[34](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR34)], the crossover is crucial for binary encoding of CNNs as it allows population members to share common building-blocks [[35](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR35), [10](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR10)]. The multi-objective search often employs the *non-dominated sorting*, known from, e.g., the NSGA-II algorithm [[36](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR36)], which enables to maintain diverse trade-offs between conflicting design objectives. The evolutionary search is usually combined with *learning* because it is very inefficient to let the evolution find the weights. A candidate CNN, constructed using the information available in its genotype, is trained using common learning algorithms available in popular DNN frameworks such as TensorFlow [[19](https://link.springer.com/article/10.1007/s10710-022-09441-z#ref-CR19)]. The number of epochs and the training data size have to be carefully chosen to reduce the training time, although by doing so, the fitness score can wrongly be estimated. The CNN accuracy, which is obtained using test data, is interpreted as the fitness score.