# Software Verification and Validation of Automated Machine Learning and Reinforcemente Learning

Hamdi Abed

Software verification and validation course

2019

# Content

## Automated Machine Learning (AutoML)

* Machine Learning overview
* Reinforcement Learning
* AutoML from a neural architecture search perspective

## Software verification and validation of AutoML

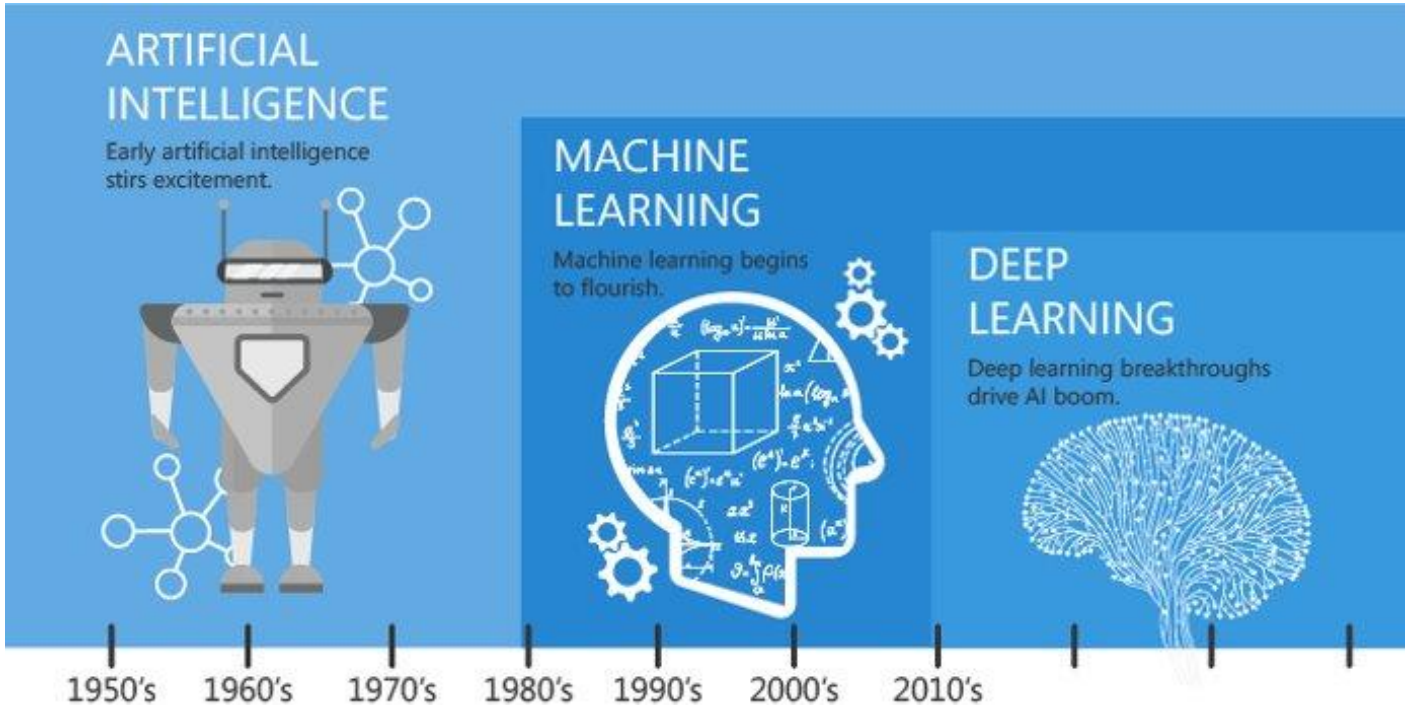
* AutoML verification techniques
* AutoML validation methods

## Conclusion

# Machine Learning introduction

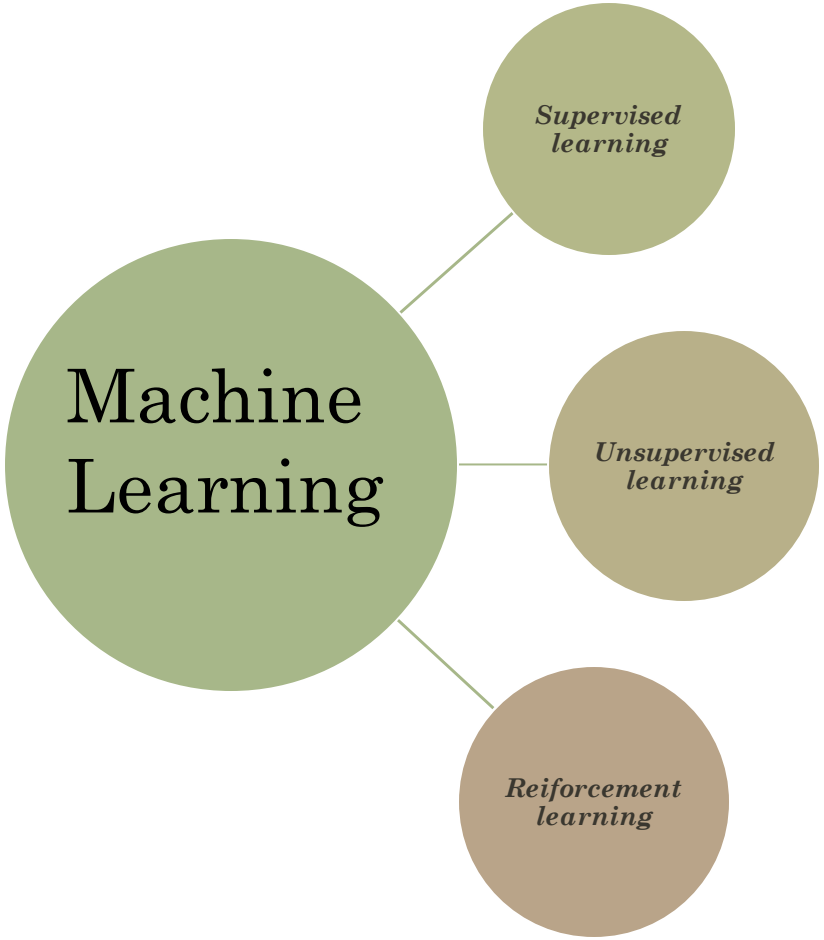
- What makes an algorithm a machine learning algorithm?
  - Pattern recognition
  - Learning from data
- Applications of ML:
  - Computer vision
    - Image classification
    - Segmentation
    - Object detection
  - Natural language Processing (NLP):
    - Text processing
    - Chatbots
  - Seq2Seq modeling

# Machine Learning history

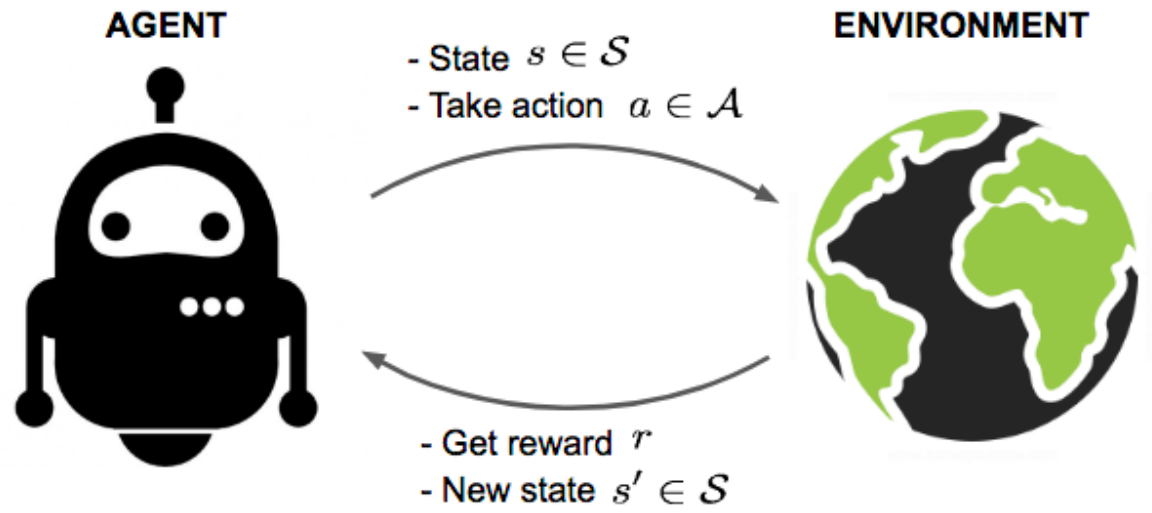



Source: https://hackernoon.com/difference-between-artificial-intelligence-machine-learning-and-deep-learning-1pcv3zeg

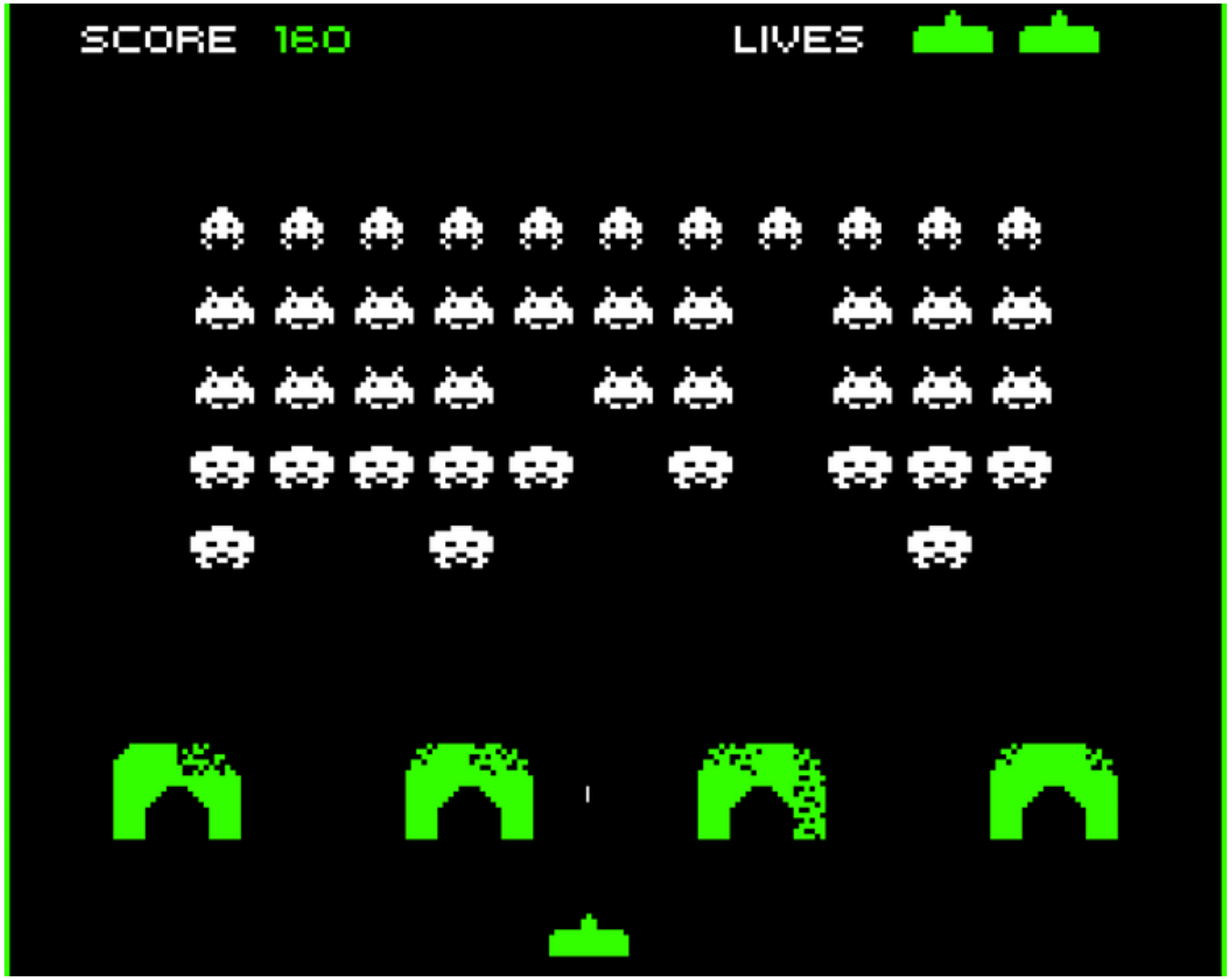
# Machine Learning types



# Reinforcement Learning

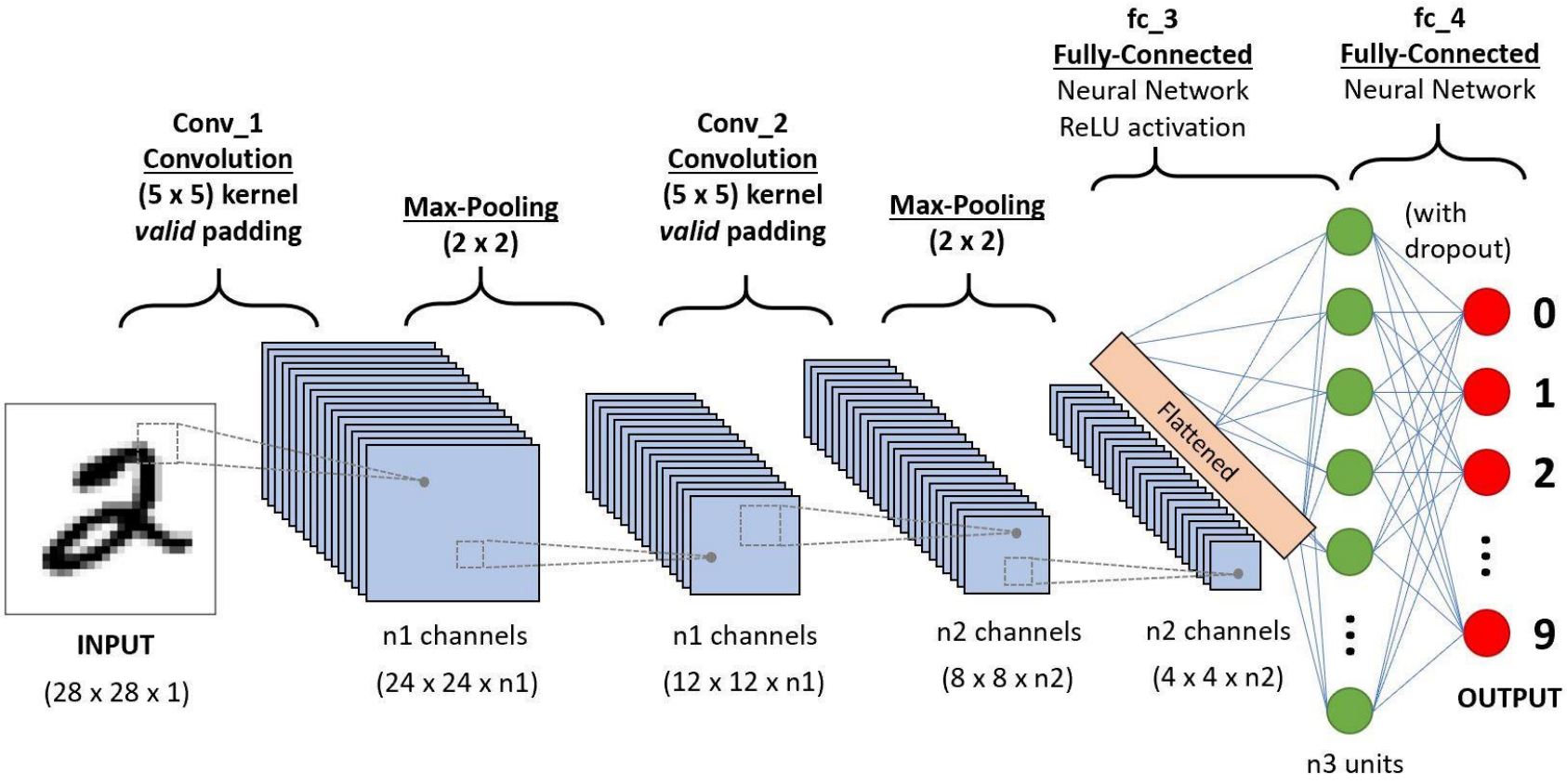



Source: https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html




# RL example

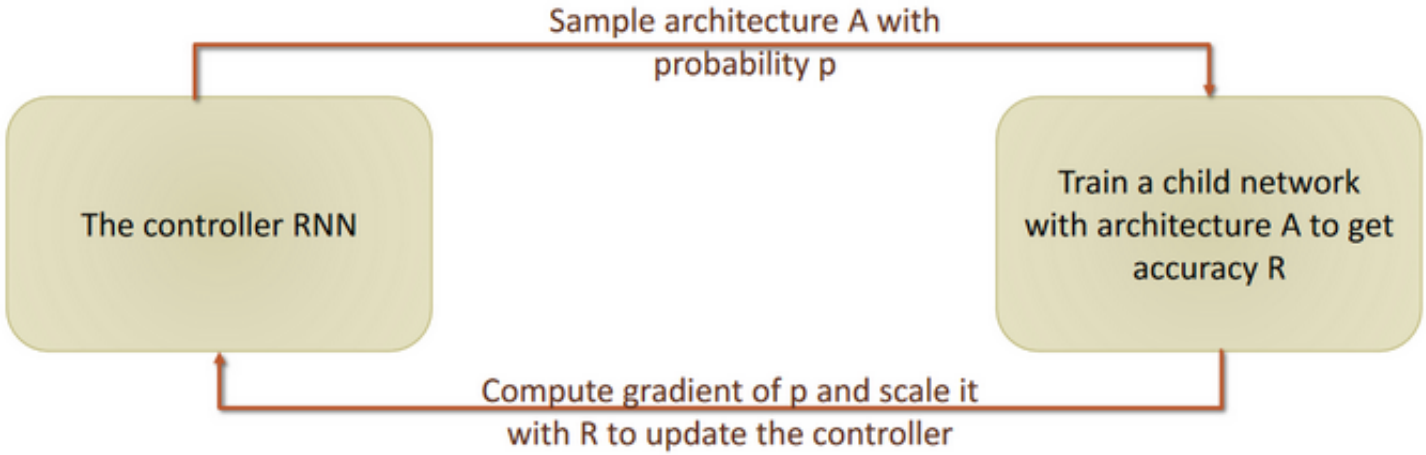
# CNN architectuer



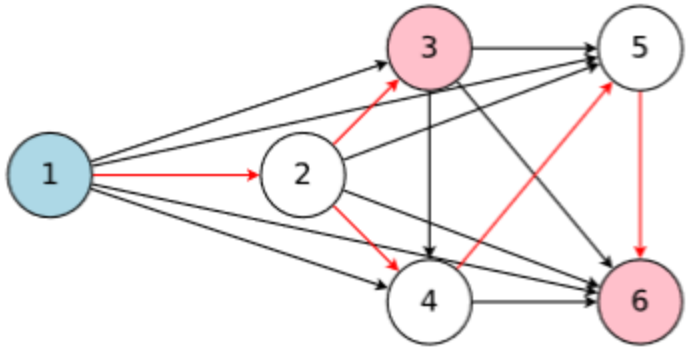

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

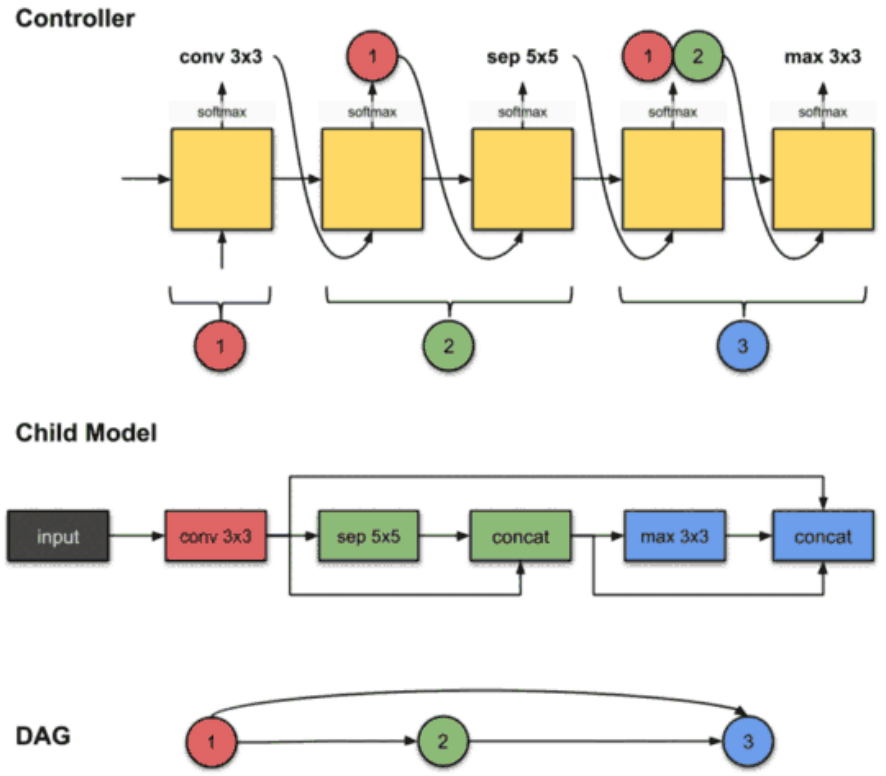
# Neural architecture search approach

- Goal:
  - To find the best convolutional neural network architecture.
  - Better accuracy , better the architecture
- Model description:
  - Controller: LSTM Cell
  - Child: CNN network
- Searching with RL is based on MDP.



# Neural network architecture search



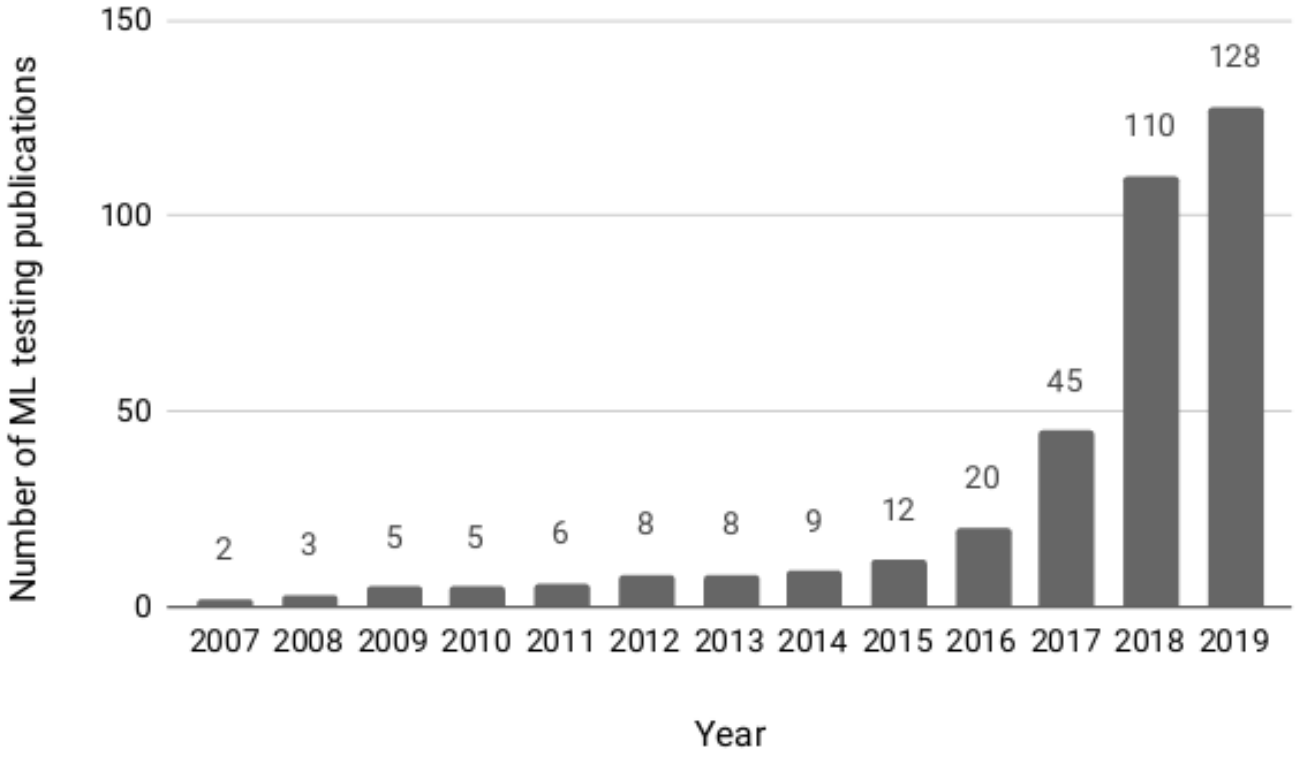




Source: https://medium.com/@SmartLabAI/introduction-to-neural-architecture-search-reinforcement-learning-approach-55604772f173

# Verification of AutoML (Model's specifications)

- As part of AI, AutoML is subjected to ISO/IEC JTC 1/SC 42 Artificial Intelligence standards.
- There are 29 participant countries, and 12 observants (including Hungary)
- An ML product's Specifications can be realized as follows:

| Perspective | Application on NAS model |
|---|---|
| Identifying the stakeholders | -Sucessful communication between various stakeholders. |
| Goals and Features | -Design for user value, not ML/AI.<br>-Leave room for experimentation. |
| Product targets / KPIs | -Set realistic and relevant goals.<br>-Use models that can be evaluated. |
| Product constraints | -Clean and useful data for training.<br>-Complexity measures. |



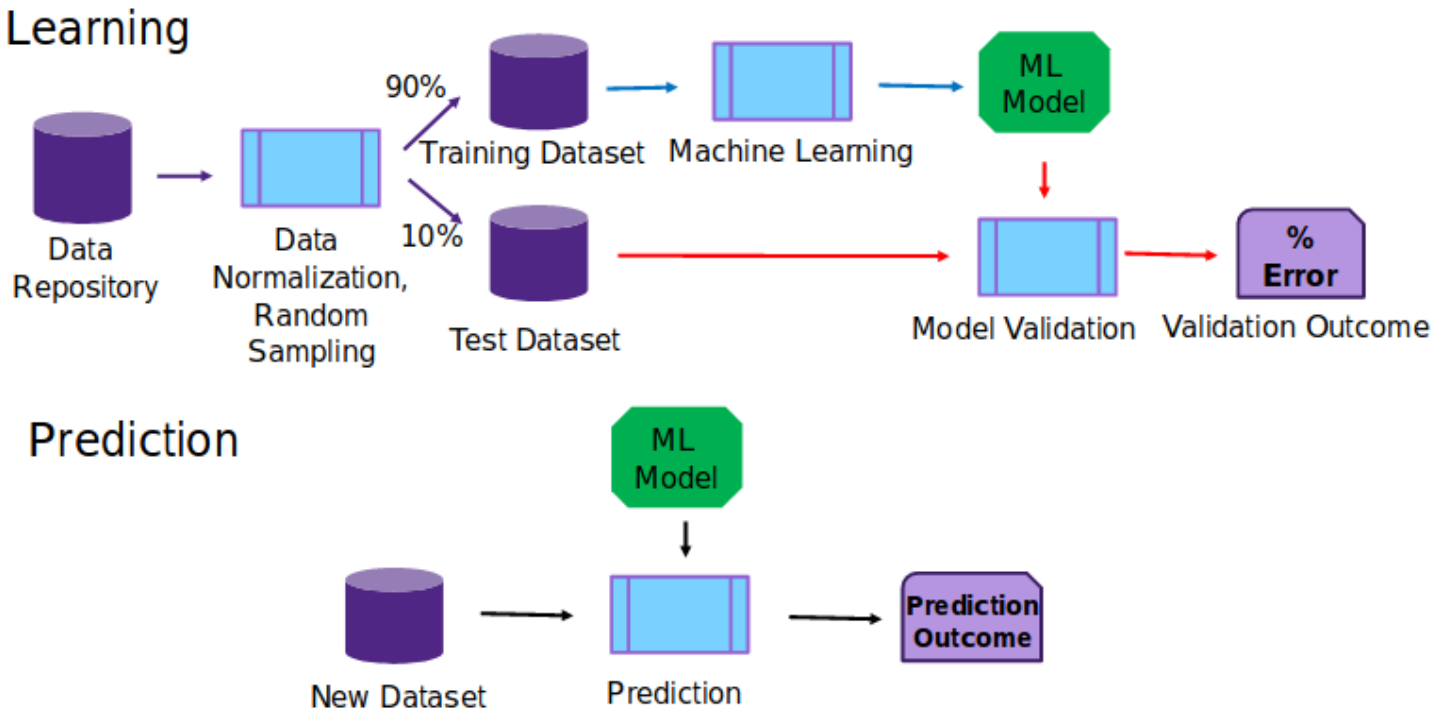

# Publications on verification of ML

Source:
https://arxiv.org/pdf/1906.10742.pdf

# Software Verification in AutoML

- The verification of AutoML software is part of Machine Learning verification.
- We need to verify these methods to insure a rubost and stable model:
  - Verification of a supervised machine learning algorithm (related to the child network).
    - Model validation
    - Fault detection and Debugging.
    - Model abstraction and interoperability.
    - Efficient designer to improve productivity.
  - Verification of reinfocement learning mehtod.(Related to the controller)
    - Platform assumptions.
    - Data assumptions.
    - Environment assumptions.
    - Machine learning algorithm assumptions.

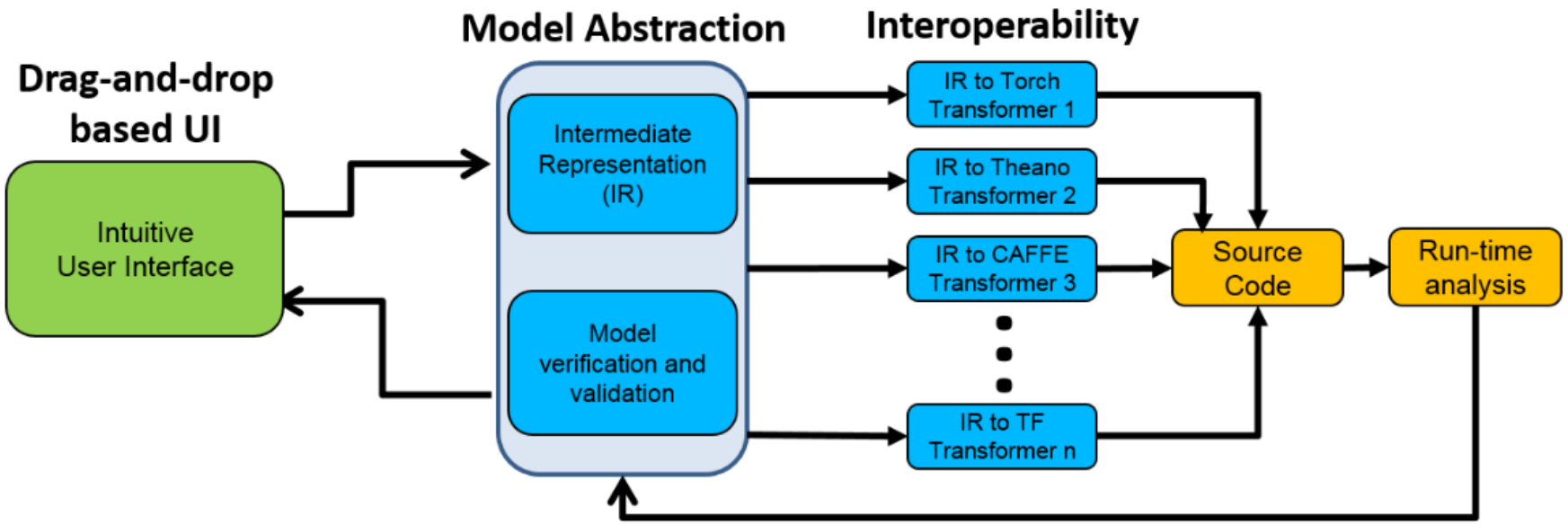
# Verification of Supervised learning




- What we automate is the ML model in this case.
- The train/test split is variable, and often we split the test set into test and validation set
- The ML model here is a convolutional neural network for classification task.
- The metrics we use is the accuracy of classification.

Source:
https://www.cs.utexas.edu/users/hunt/FMCAD/FMCAD16/slides/tutorial1.pdf

# Verification of supervised learning




Source: file:///home/hamdiabed/Downloads/DARVIZ_%20Deep%20Abstract%20Representation.pdf

# Verification of NAS child

| Verification method | Application |
| --- | --- |
| Model validation | - Correct hyperparameters selection. |
| Fault detection and debugging | - Large data causes slow training of models.<br>- Real-time fault detection can reduce training time. |
| Model Abstraction and Interoperability | -Model interpretation between different platforms. Ex:(CAFFE models in TensorFlow) |
| Efficient designer to imprtove productivity | -Building a design which can generate code in any realization platform.<br>-Designs should have steep learning curves.<br>-Good set of features selection. |

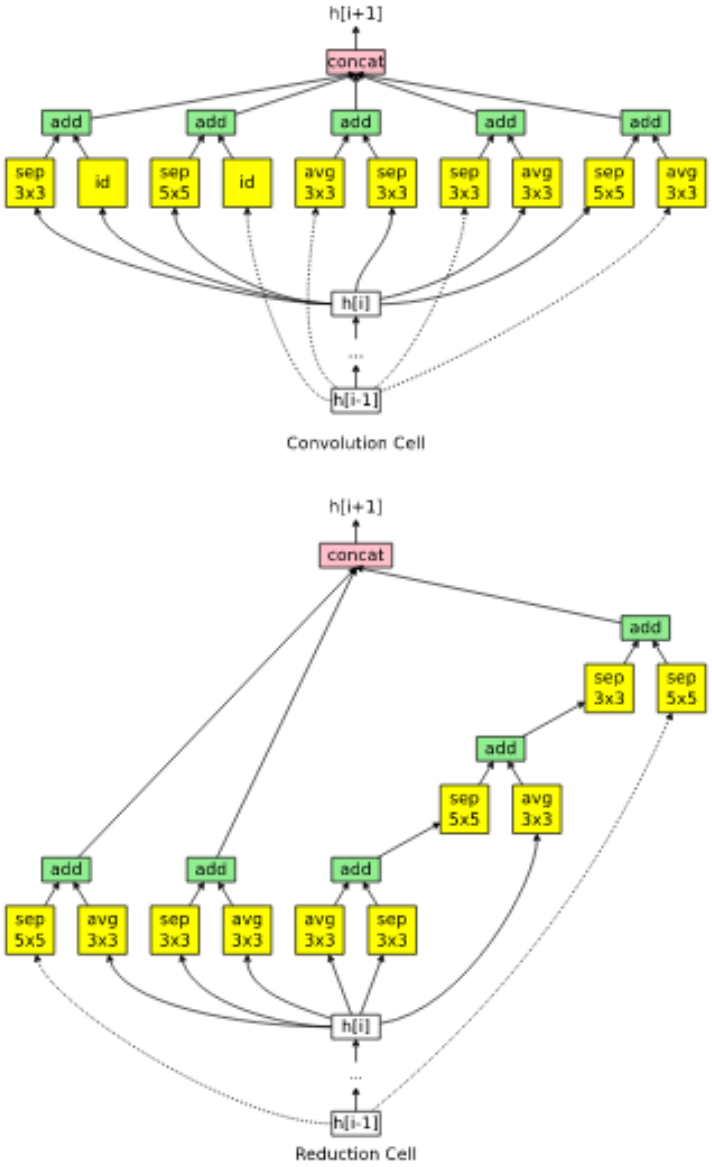
# Verification of NAS child network

- Child's hyperparameters:
  - Number of layers = 2
  - Number of cells per layer = 5
  - Output filters = 24
  - Batch size = 32
  - Learning rate = 0.1
  - Learning rate decay = 0.1
- Model was based on TensorFlow platform and trained on CiFAR-10 dataset.
- It took 1 GPU day to process this search.

```
class MicroChild(Model):
  def __init__(self,
               images,
               labels,
               use_aux_heads=False,
               cutout_size=None,
               fixed_arc=None,
               num_layers=2,
               num_cells=5,
               out_filters=24,
               keep_prob=1.0,
               drop_path_keep_prob=None,
               batch_size=32,
               clip_mode=None,
               grad_bound=None,
               l2_reg=1e-4,
               lr_init=0.1,
               lr_dec_start=0,
               lr_dec_every=10000,
               lr_dec_rate=0.1,
               lr_cosine=False,
               lr_max=None,
               lr_min=None,
               lr_T_0=None,
               lr_T_mul=None,
               num_epochs=None,
               optim_algo=None,
               sync_replicas=False,
               num_aggregate=None,
               num_replicas=None,
               data_format="NHWC",
               name="child",
               **kwargs
              ):
```

# Verification of NAS child network

- Child's hyperparameters:
  - Number of layers = 2
  - Number of cells per layer = 5
  - Output filters = 24
  - Batch size = 32
  - Learning rate = 0.1
  - Learning rate decay = 0.1



# Verification of RL (NAS controller)

- There are four main assumptions for RL verification:

| Assumption | Application |
|---|---|
| Assumption of RL environment | -What is the environment we will work within. |
| Assumptions of platform | -Do we need more components!<br>-Components failure assumptions.<br>-Componenets quality and maintainence. (fidelity assumptiosn) |
| Assumptions of data | -Independent data, and well-distributed.<br>-Proximity!<br>-Adversarial behaviour |
| Assumptions of RL machine learning algorithm | -Describes the functionality of the data.<br>-Accurately modelling the data.<br>-Always optimize towards the global optimum. |

| | Environmental assumptions | Platform assumptions | Data assumptions | Algorithm assumptions |
|---|---|---|---|---|
| **Environmental assumptions** | × | | | |
| **Platform assumptions** | Platform assumptions can be based on assumptions on the environment, e.g. a sensor has a 95% accuracy at temperatures between x and y. | × | | |
| **Data assumptions** | The training data is assumed to adhere to the assumptions and constraints of the environment. | The data might be captured by the platform sensors, thus being affected by any fidelity or failure assumptions. | × | |
| **Algorithm assumptions** | The chosen algorithm is assumed to model the environment of the system well. | The algorithm used assumes the availability and accuracy of inputs. | Properties of the algorithm such as convergence depend on the data it sees. | × |

# Verification of RL (NAS controller)

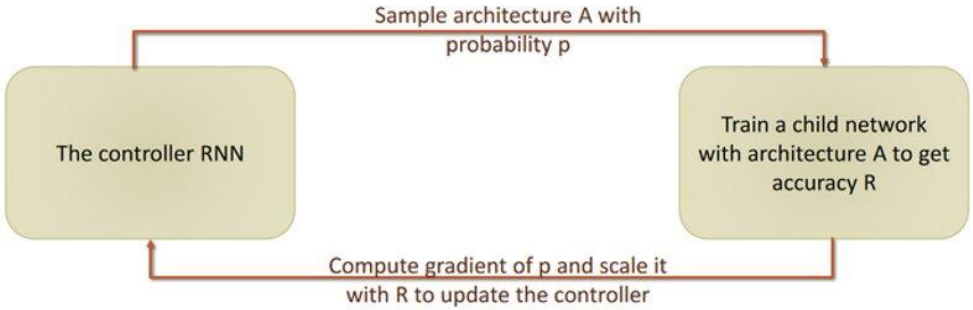
How does the RL verification assumptions influence each other

# Verification of RL

- The formal verification is mathimatically proving that the system satisfies the specifications.
- But what are the specifications!
- To ensure that our predicted output of the system is close to the target output.
- There are two main categories of RL verification:
  - Offline verification: which can be done after the training is completed.
  - Online verification (Runtime verification): which can be done during the training.

# Verification of RL

- State: The current architecture that is trained.
- Action: the next architecture to be trained.
- Environment: All possible architectures.
- Aim: is to find the best architecture for classification task.
- Reward: bigger reward for more accurate architecture.



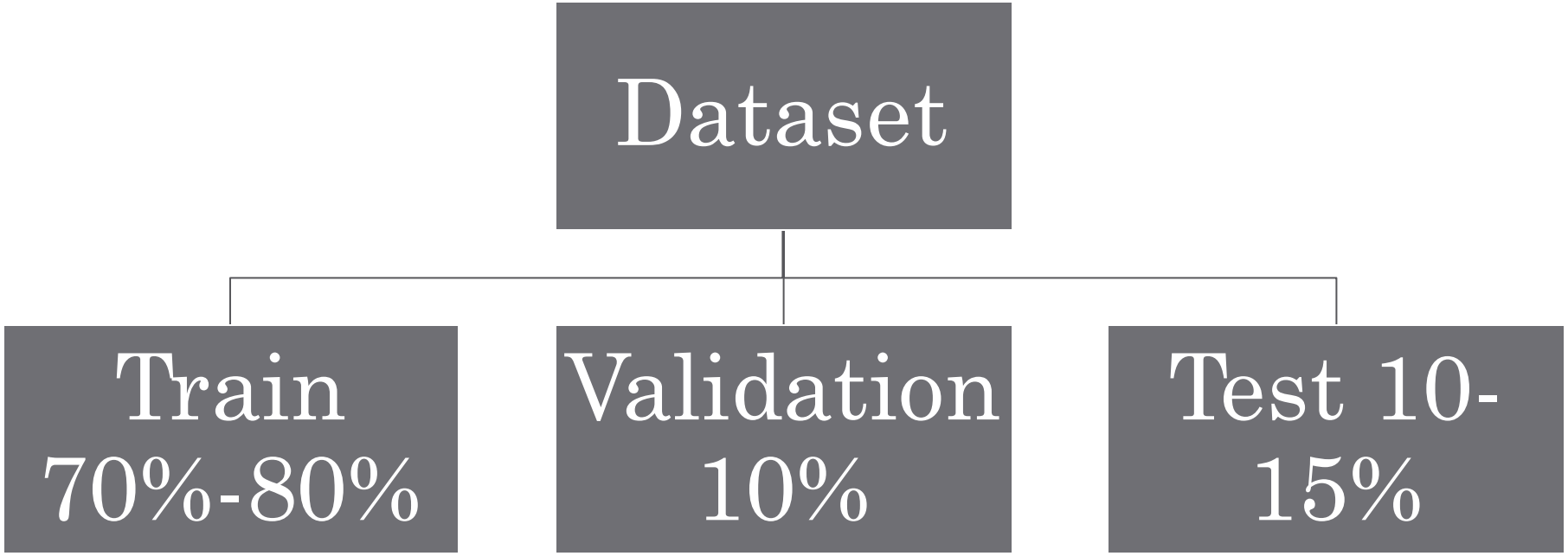
# Offline verification of RL

- State to action mappings:
  - After one controller epoch, choose the most accurate arch.
- Action sequences:
  - Give a higher probability to previousely better perofroming archs, and gice a smaller probabilty to the others.
- RL algorithm characteristicss should be independent of the data:
  - Does using policy gradient for optimization actually works?
- Model validation:
  - Will be expained later.

# Online verification of RL

- State to action mappings:
  - At each step of the controller verify that we choose the highest accuracy arch.
- Action sequences:
  - Is the runtime verification of action selection.
- Monitoring of our assumptions:
  - Continous check-ups of the environment, should be done during the pass of training set.
- Input instance in training data:
  - Avoid the actions that are not useful during training.

# AutoML Validation



# AutoML Validation

- Validation is used during the training to avoid overfitting to data.
- Test set is different from validation set.
- Both validation and test are used to Validate a machine learning model.
- There are various validation techniques.
- In classification problems:
  - Binary validation, when we have binary classification problem.
  - K-Fold cross validation , for multiclass tasks.
- In our ENAS model, validation accuracy was the measurement of the architecture's superiority.

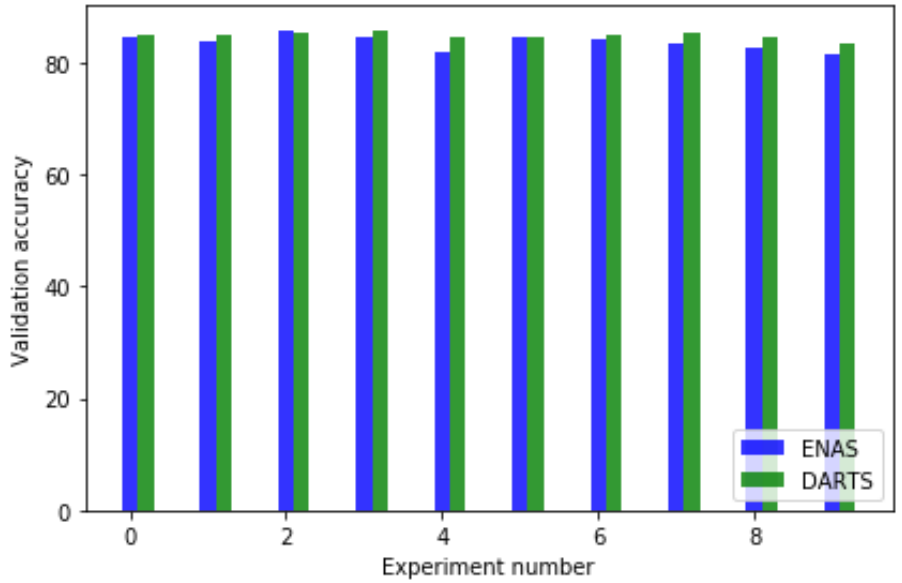
# AutoML Validation

- Types of ML validation techniques:

| Validation technique | Method |
|---|---|
| Resubstitution | - Error is evaluated based on outcome vs. actual value from the same training data set<br>- Data is not splitted. |
| Holdout | - In this case data is train-test splitted.<br>- Better results than the first method |
| K-Fold cross validation | - An iterative method that uses the data on batches and computes the average error for each batch |
| Leave-One-Out validation | - All data is used for training exept one instance for testing, this instance is sampled differently every epoch. |
| Random sub-sampling | - Sample a subset of the data set randomly at each iteration. |
| Bootstrapping | - The training dataset is sampled, and what is left is for testing. |

# AutoML Validation

- Accuracy of ENAS



# Conclusion

- AutoML formal verification can be achieved with online and offline verification techniques.
- In Online verification, metrics are measured during the training of the model, meanwhile in offline verification, the measurement is when the training is completed.
- Validation of AutoML is similar to machine learning validation, and it uses different subset of the dataset than the training set.

# References


- Paper of ENAS: https://arxiv.org/pdf/1802.03268.pdf
- Code of ENAS: https://github.com/melodyguan/enas
- Machine learning testing: https://arxiv.org/pdf/1906.10742.pdf
- Machine learning and Formal methods: https://drops.dagstuhl.de/opus/volltexte/2018/8430/pdf/dagrep_v007_i008_p055_17351.pdf
- Machine learning for verification: https://www.cs.utexas.edu/users/hunt/FMCAD/FMCAD16/slides/tutorial1.pdf
- Challenges in the verification of RL: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20170007190.pdf