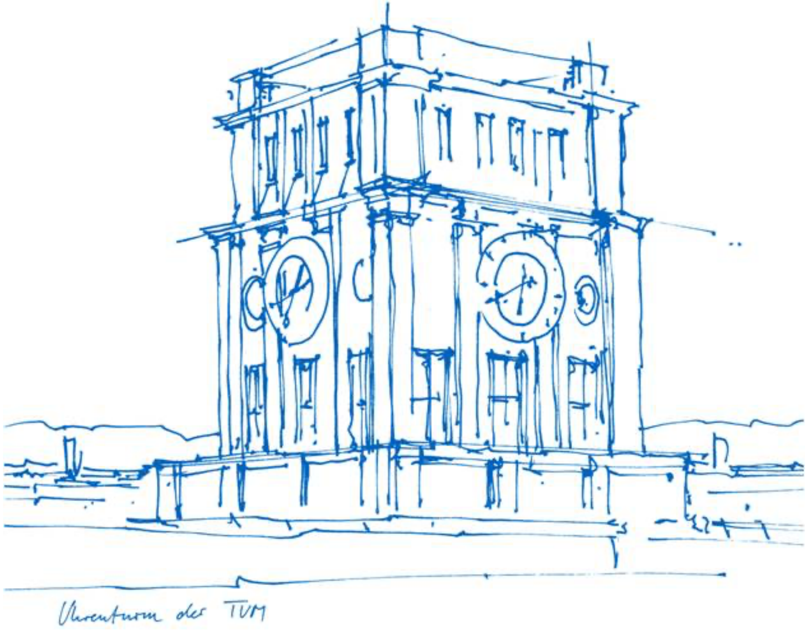
Neural Architecture Search for Automated Machine Learning Deployment on Extreme Edge Devices

Philipp van Kempen (TUM)

Supervisor: Alexander Hoffman

Wissenschaftliches Seminar Realzeit-Computersysteme

Winter Term 2020



Content

- Motivation
- Introduction of Core Concepts

- State of the Art
- Comparisons

- Conclusions
- Outlook

Motivation (1)

- Machine learning and AI are omnipresent in our lives
- Neural networks on…
  - …Servers/Computers/Smartphones
  - …Microcontrollers (MCUs)?

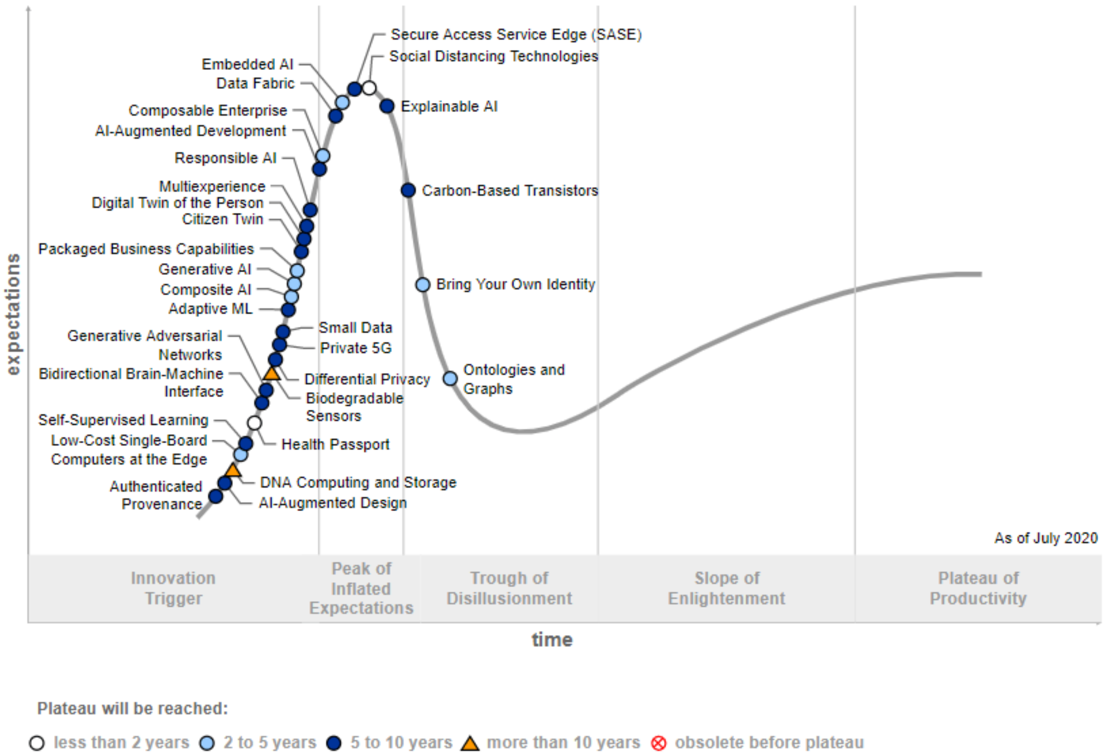


Figure: 2020 Gartner Hype Cycle

Motivation (2)

- **TinyML:** Machine Learning on extreme edge devices

- Several constraints:
  - Computational Power
  - Memory Size (Flash, RAM)
  - Battery Life
  - Size
  - …

→ How to design neural network models suitable to run on Embedded Systems?

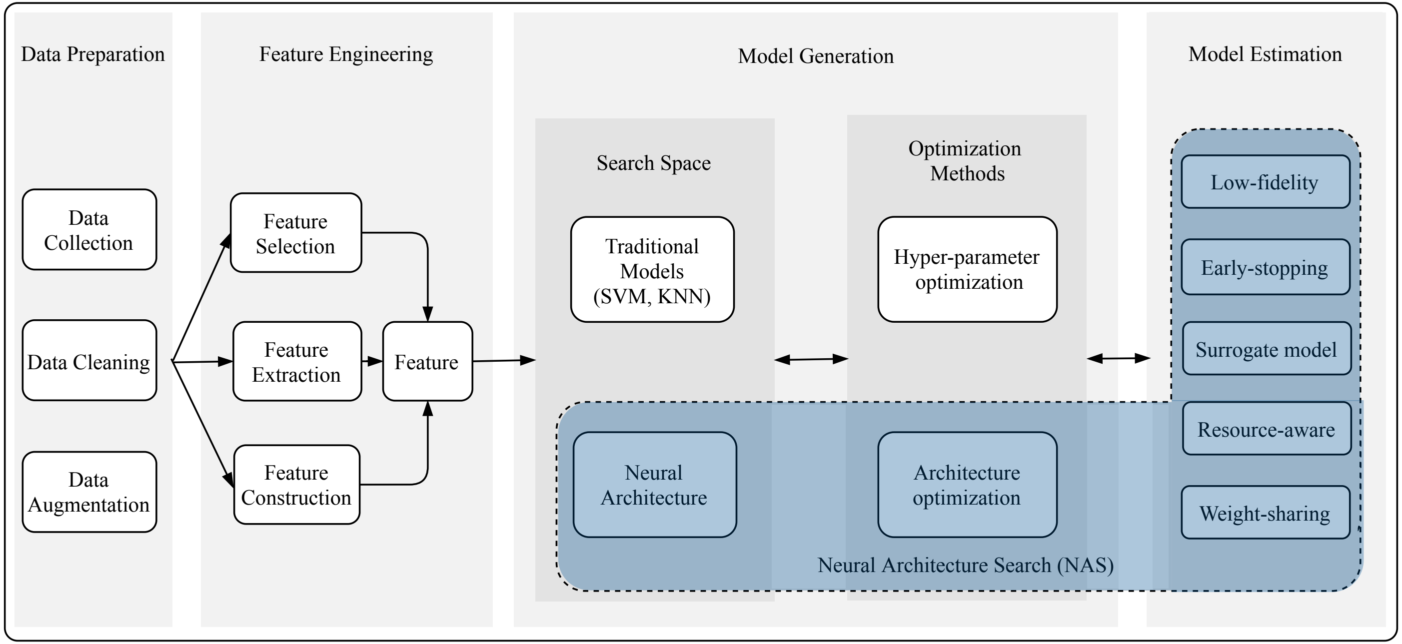
Neural Architecture Search (NAS)

- Essential part of AutoML [1]
- Reduces engineering effort for developing Neural network models
- Algorithms searching for the best neural network architecture

- **Objectives:**
  – Accuracy
  – Latency
  – Model Size → static/dynamic memory usage
  – Power Constraints

\- Bad choice for bigger network architectures due to long training times (Servers, Computers)
\+ Suitable for Embedded Devices (Smartphones, Microcontrollers) → Smaller models

Role of NAS in AutoML Flow



Source: HE, Xin; ZHAO, Kaiyong; CHU, Xiaowen. AutoML: A Survey of the State-of-the-Art. *Knowledge-Based Systems*, 2020, S. 106622.

Model Compression

- Reducing Model Size and Complexity is essential for Embedded Devices
- **Often:** Compromises

- **Techniques:**
  - Weight-Sharing/Parameter-Sharing
  - Quantization
  - Pruning
  - ...

Machine Learning Frameworks

Papers	Frameworks				
	TFLite (Micro)	CMSIS-NN	MircoTVM	Pytorch	Custom
MCUNet	•	•	•		•
µNAS	•	•			
SpArSe	N/A	N/A	N/A	N/A	N/A
MicroNet	•	•			
Once-for-All				•	

• for comparisons only **• for evaluations** N/A: no details available

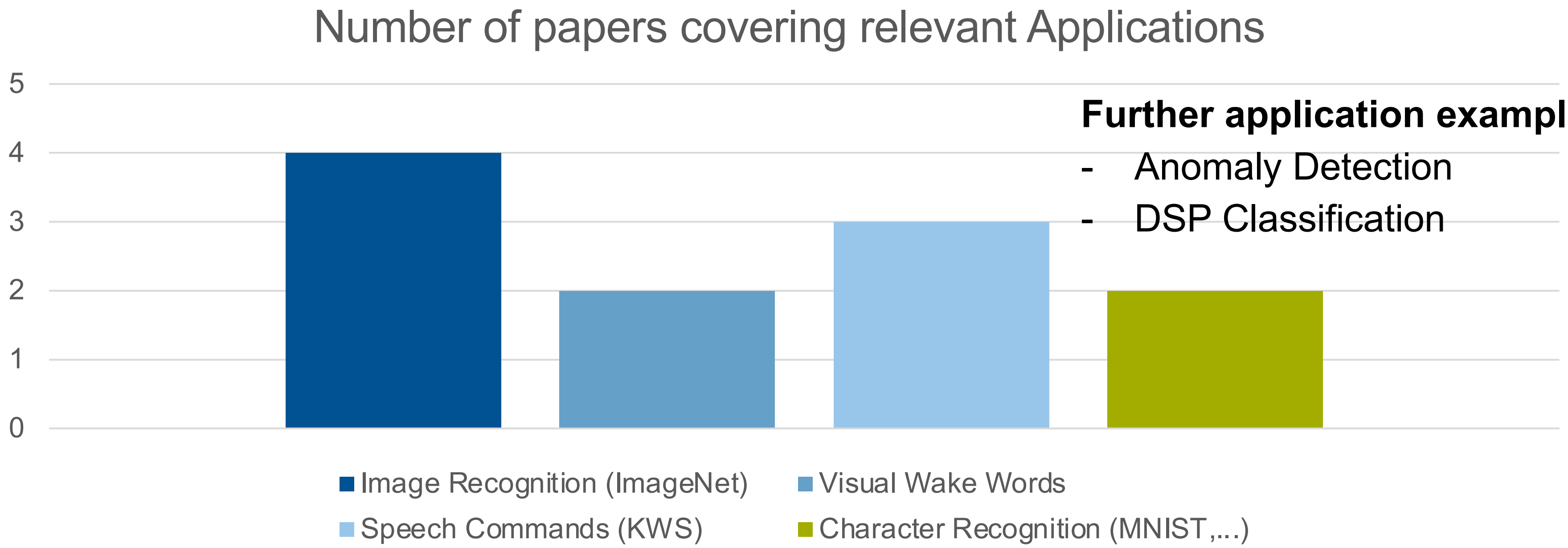
Content

- Motivation
- Introduction to Core Concepts

- State of the Art
- Comparisons

- Conclusions
- Outlook

Datasets and Target Applications



Further application examples:

- Anomaly Detection
- DSP Classification

Paper I: *MCUNet*

"MCUNet: Tiny Deep Learning on IoT Devices", 2020 [2]

Proposal:

- System-model co-design framework made of 2 core components:
  - *TinyNAS*: Search algorithm suitable for MCUs
  - *TinyEngine*: Lightweight inference engine based on code generation

- TinyNAS achieves more promising results with the larger available search space due to the large amount of optimizations applied in the TinyEngine
- TinyNAS follows two-step approach to take resource constraints into account

Paper II: *μNAS*

"μNAS: Constrained Neural Architecture Search for Microcontrollers", 2020 [3]

Proposal:

- Approach which generates mid-tier MCU-level networks
  → primarily intended for image classification tasks
- Multi-objective optimization taking RAM-size, persistent storage and processor speed into account
- Design requirements:
  - Highly granular search space
  - Accurate resource use computation

Paper III: SpArSe

“SpArSe: Sparse architecture search for CNNs on resource-constrained microcontrollers”, 2019 [4]

Proposal:

- AutoML techniques can generate Convolutional Neural Networks (CNNs) for memory-constrained MCUs which also generalize well
- Bayesian optimization of three objective functions
  - Validation Accuracy
  - Model Size
  - Working Memory
- Network morphism with random scalarizations follows model size compression via pruning

Paper IV: MicroNets

"MicroNets: Neural Network Architectures for Deploying TinyML Applications on Commodity Microcontrollers", 2020 [5]

Proposal:

- Studied properties of NAS search spaces for MCU model design
  → correlation between the model latency and the model operation count
- Latency/energy model as a prerequisite to apply NAS
- Models targeting three different size-classes: S/M/L

Paper V: Once-for-All

"Once for all: Train one network and specialize it for efficient deployment", 2019 [6]

Proposal:

- Approach to design optimal networks suitable for a wide range of devices
- Decoupling Training and Search steps

  → No additional expensive training time in the deployment stage
- Exploiting weight sharing and progressive shrinking

Comparison (1)

- **Available Metrics:** Model accuracy, program size, SRAM usage, inference latency, MACs/FLOPs, training time, search cost, energy consumption → *Multi-Objective optimization problem*

- *“**MCUNet** vs. TFLite”: 3.4× less* Memory usage 1.7-3.3× shorter Inference time
  → TinyEngine increases feasible search space

- Image Classification with **μNAS**: up to 4.8% higher accuracy
  4-13× smaller memory footprint vs. 900× less MACs
  μNAS outperforms SpArSe in Character Classification

- **SpArSe:** more accurate and up to 4.35× smaller models

Comparison (2)

- *“**MicroNets** vs. MCUNet”:* Outperforms MCUNet in KWS tasks (Keyword Spotting)
- **Once-for-All:** Promising results for many datasets and target architectures
  → Can reduce design effort by a high degree

Content

- Motivation
- Introduction to Core Concepts

- State of the Art
- Comparisons

- Conclusion
- Outlook

Conclusion

- **AutoML/NAS for MCUs**: Highly interesting research filed

- **State of the Art**:
  1. Co-Design of NAS algorithm with Framework/Inference Engine yields the best results (→ MCUNet)
  2. Model Compression algorithms are playing a large role
  3. Model op count and Inference latency have a linear relation → Cost function without training
  4. Large Super-networks help to extract smaller sub-networks by sharing weights/parameters

Future Work

- **Further thoughts**:
  - Open-sourced implementations allow further research on hybrid approaches

- **Lightweight Machine Learning Research Topics**:

1. Mixed low bitwidth quantization
2. Standard convolutions instead of depth-wise convolutions

References

[1] HE, Xin; ZHAO, Kaiyong; CHU, Xiaowen. AutoML: A Survey of the State-of-the-Art. *Knowledge-Based Systems*, 2020, S. 106622.

[2] J. Lin, W.-M. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, “MCUNet: Tiny Deep Learning on IoT Devices,” no. NeurIPS, pp. 1–12, 2020.

[3] E. Liberis, Ł. Dudziak, and N. D. Lane, “μNAS: Constrained Neural Architecture Search for Microcontrollers,” 2020.

[4] I. Fedorov, R. P. Adams, M. Mattina, and P. N. Whatmough, “SpArSe: Sparse architecture search for CNNs on resource-constrained microcontrollers,” *arXiv*, pp. 1–26, 2019.

[5] C. Banbury, C. Zhou, I. Fedorov, R. M. Navarro, U. Thakker, D. Gope, V. J. Reddi, M. Mattina, and P. N. Whatmough, “MicroNets: Neural Network Architectures for Deploying TinyML Applications on Commodity Microcontrollers,” 2020.

[6] H. Cai, C. Gan, and S. Han, “Once for all: Train one network and specialize it for efficient deployment,” *arXiv*, pp. 1–15, 2019.

Questions?