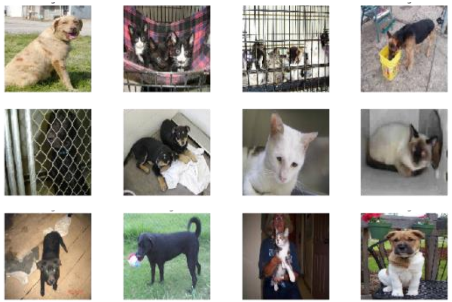
Automatisierte Augmentationen für Active Learning

I705 Forschungs- und Entwicklungsprojekt / -seminar

Prof. Dr. Maik Thiele | Colin Simon, Serhiy Bolkun, Kevin Kirsten

Conventional (Passive) Learning

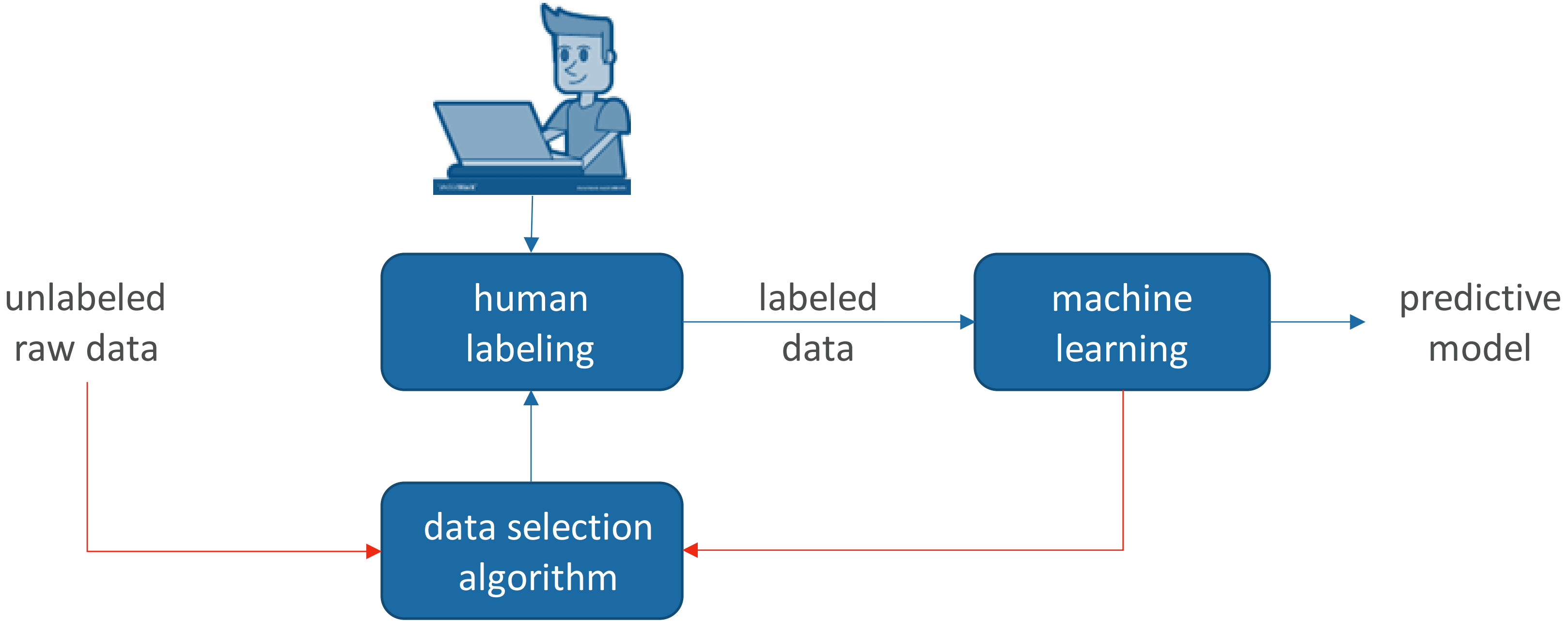
unlabeled raw data → **human labeling** → labeled data → **machine learning** → predictive model



dog

cat

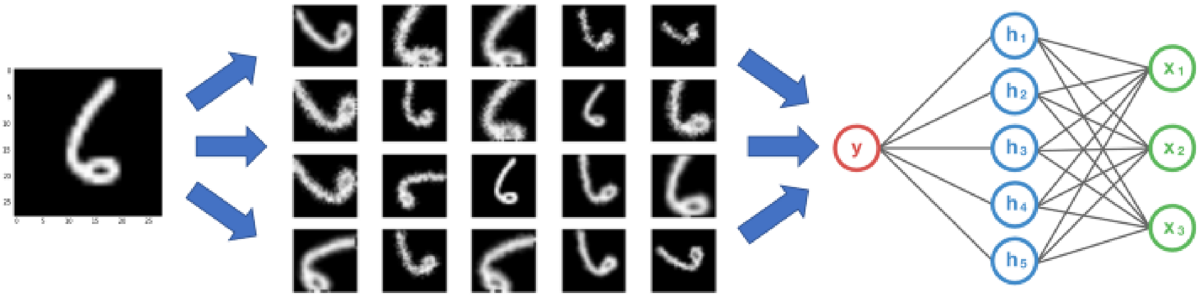
Active Learning



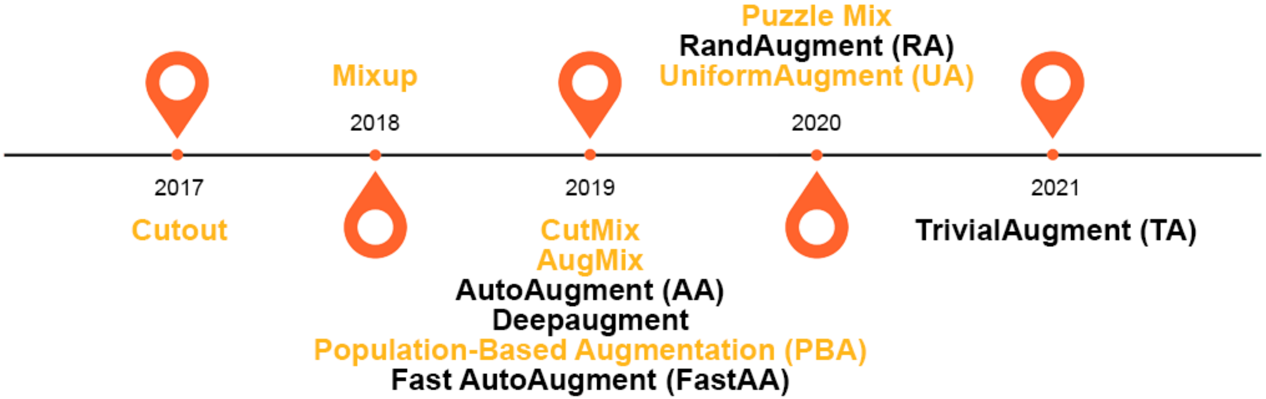
Goal: machine automatically and adaptively selects most informative data for labeling
→ collect best data at minimal cost

Data Augmentation Strategien

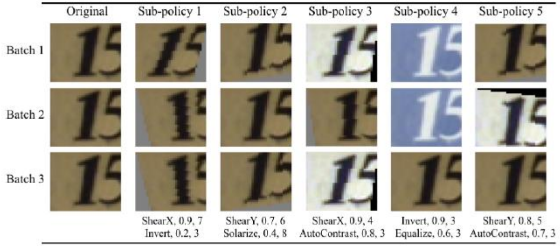
Image Data Augmentation Strategien



Your neural network is only as good as the data you feed it.



AutoAugment



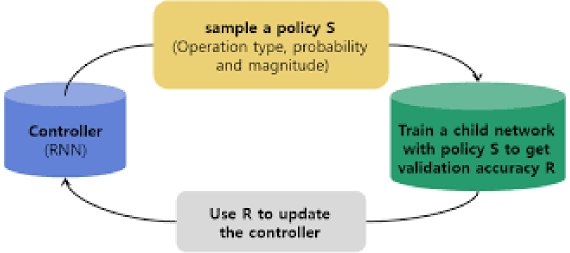
Ablaufplan:

(1) Policy Suche

- Suche Policies mithilfe des Reinforcement Learning Loop
- Speichere die Top 5 Policies

(2) Policy Anwendung

- Teile deine Training Batches auf in Mini Batches
- Wende zufällig eine der Sub Policy auf dein Mini-Batch an



Search space:

$$(16\ op \times 10\ m \times 11\ pr)^{2 \times 5} \approx 2.9 \times 10^{32}$$

op – number of operations

m - magnitude range (uniform spacing)

pr – probability range (uniform spacing)

	Operation 1	Operation 2
Sub-policy 0	(Invert,0.1,7)	(Contrast,0.2,6)
Sub-policy 1	(Rotate,0.7,2)	(TranslateX,0.3,9)
Sub-policy 2	(Sharpness,0.8,1)	(Sharpness,0.9,3)
Sub-policy 3	(ShearY,0.5,8)	(TranslateY,0.7,9)
Sub-policy 4	(AutoContrast,0.5,8)	(Equalize,0.9,2)

Google AI

Deepaugment

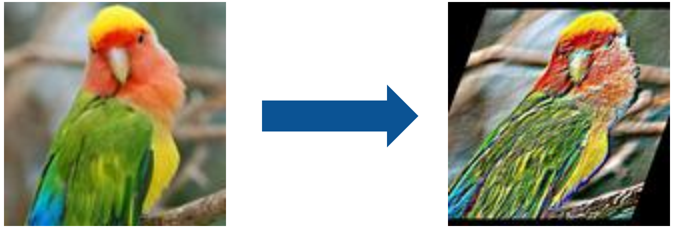
5X

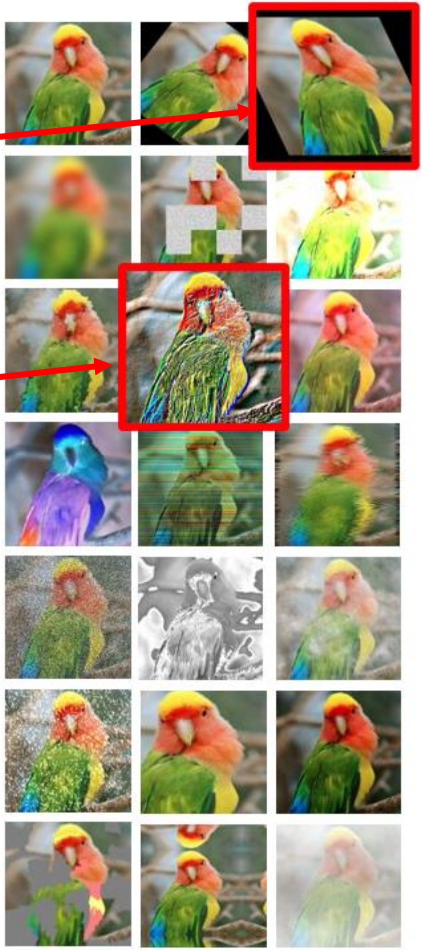
First aug. type — Shear

First aug. magnitude — 12

Second aug. type — Emboss

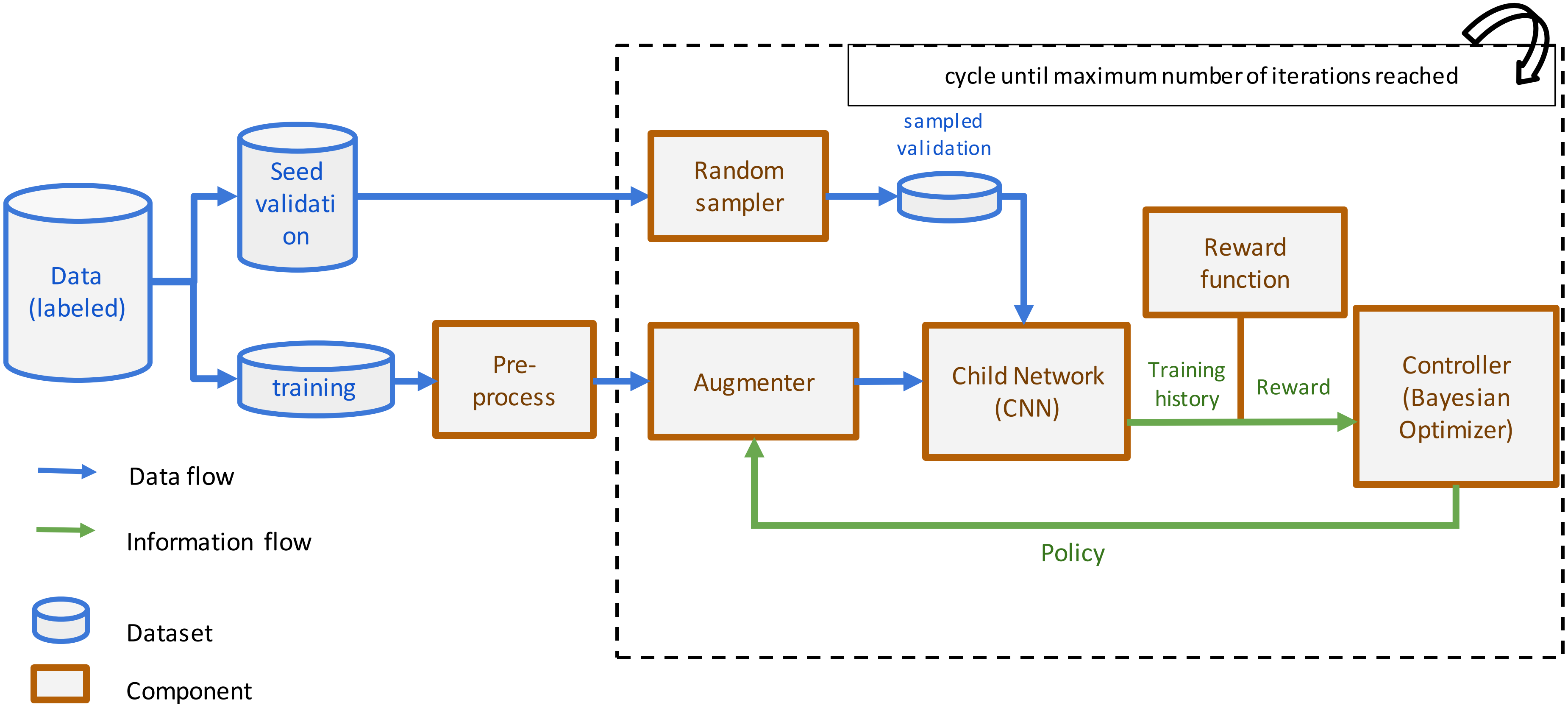
Second aug. magnitude — 9





B. Özmen, 'Deepaugment', 2019 (https://github.com/barisozmen/deepaugment)

Deepaugment - Pipeline for learning policies

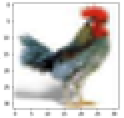


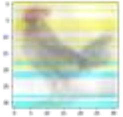
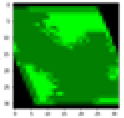
Deepaugment - Best und schlechteste Strategie CIFAR-10

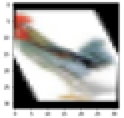
aug1-type	aug1-magnitude	aug2-type	aug2-magnitude	portion	estimated-accuracy-boost (%)
horizontal-flip	0.90	gamma-contrast	0.87	0.93	**8.3**
crop	0.90	invert	0.02	0.62	**7.4**
translate-x	0.33	fog	0.46	0.46	**7.1**
:	:	:	:	:	**:**
brighten	0.30	shear	0.19	0.67	**-17.1**
dropout	0.34	shear	0.05	0.57	**-17.4**
shear	0.34	shear	0.21	0.67	**-30.8**

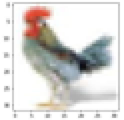
Best 3

Worst 3







original

Fast AutoAugment

- **Motivation**:
  - ➢ Policy search time for AutoAugment too large
  - ➢ Use **Bayesian optimization** techniques to speed up the search process.

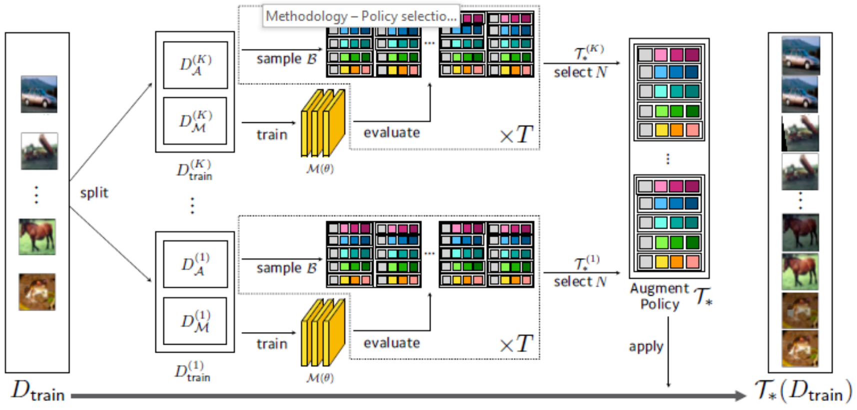
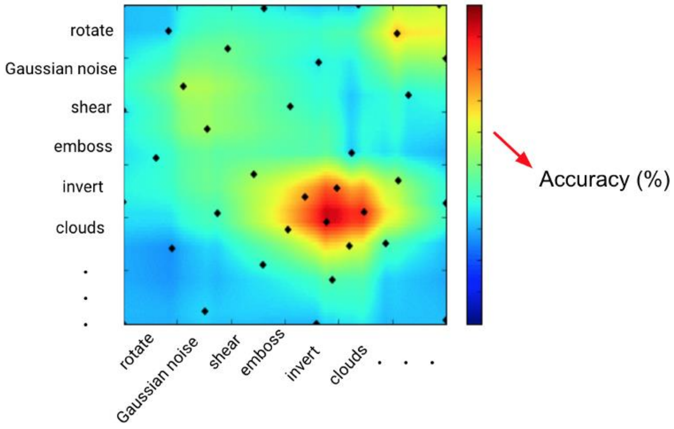


Figure 2: An overall procedure of augmentation search by Fast AutoAugment algorithm. For exploration, the proposed method splits the train dataset D_{train} into K-folds, which consists of two datasets $D_{\mathcal{M}}^{(k)}$ and $D_{\mathcal{A}}^{(k)}$. Then model parameter θ is trained in parallel on each $D_{\mathcal{M}}^{(k)}$. After training θ, the algorithm evaluates B bundles of augmentation policies on $D_{\mathcal{A}}$ without training θ. The top-N policies obtained from each K-fold are appended to an augmentation list $\mathcal{T}_*$.



Dataset	AutoAugment [3]	Fast AutoAugment
CIFAR-10	5000	3.5
SVHN	1000	1.5
ImageNet	15000	450

Table 1: GPU hours comparison of Fast AutoAugment with AutoAugment. We estimate computation cost with an NVIDIA Tesla V100 while AutoAugment measured computation cost in Tesla P100.

RandAugment: A drastically reduced search space

Philosophie: Data Augmentation auf finalem Modell -> bessere Anpassung an Modell- und Datensatzgröße

Daraus folgt:
- Keine Policy Suche
- Kein Proxy Model
- Stark verkleinerter Suchraum

	search space	CIFAR-10 PyramidNet	SVHN WRN	ImageNet ResNet	ImageNet E. Net-B7
Baseline	0	97.3	98.5	76.3	84.0
AA	10^{32}	98.5	98.9	77.6	84.4
Fast AA	10^{32}	98.3	98.8	77.6	-
PBA	10^{61}	98.5	98.9	-	-
RA (ours)	10^{2}	98.5	99.0	77.6	85.0

Table 1. **RandAugment matches or exceeds predictive performance of other augmentation methods with a significantly reduced search space.**

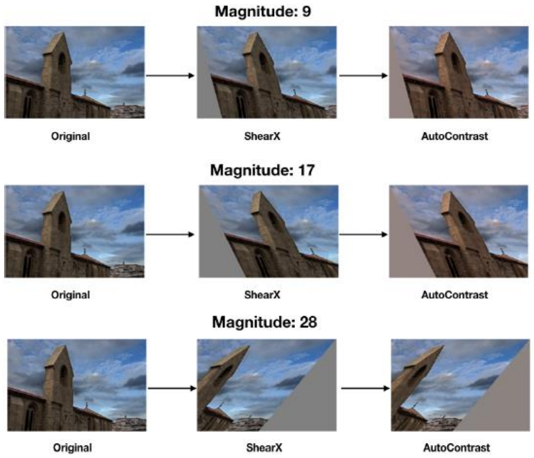


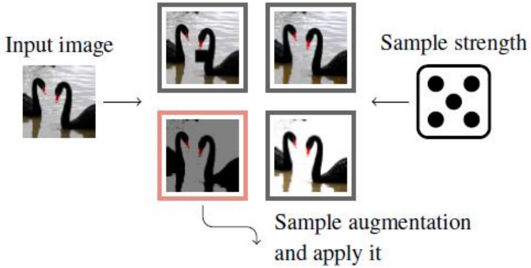
Figure 1. **Example images augmented by RandAugment.** In these examples N=2 and three magnitudes are shown corresponding to the optimal distortion magnitudes for ResNet-50, EfficientNet-B5 and EfficientNet-B7, respectively. As the distortion magnitude increases, the strength of the augmentation increases.

```
transforms = [
'Identity', 'AutoContrast', 'Equalize',
'Rotate', 'Solarize', 'Color', 'Posterize',
'Contrast', 'Brightness', 'Sharpness',
'ShearX', 'ShearY', 'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to
        apply sequentially.
    M: Magnitude for all the transformations.
"""

  sampled_ops = np.random.choice(transforms, N)
  return [(op, M) for op in sampled_ops]
```

Figure 2. Python code for RandAugment based on numpy.

TrivialAugment: a most simple baseline

Philosophie: parameterfrei und wendet nur eine einzige Augmentation an zu jedem Bild.



Algorithm 1 TrivialAugment Procedure

1: **procedure** TA(x: image)
2: Sample an augmentation a from $\mathcal{A}$
3: Sample a strength m from $\{0, \ldots, 30\}$
4: Return $a(x, m)$
5: **end procedure**

	Default	PBA	Fast AA	AA	RA	UA	TA (Wide)
CIFAR-10							
Wide-ResNet-40-2	96.16 ± .08	-	**96.4**	96.3	-	96.25	96.32 ± .05
Wide-ResNet-28-10	97.03 ± .07	**97.4**	97.3	**97.4**	97.3	97.33	**97.46** ± .06
ShakeShake-26-2x96d	97.54 ± .07	98.0	98.0	98.0	98.0	98.10	**98.21** ± .06
PyramidNet	97.95 ± .05	**98.5**	**98.5**	98.3	**98.5**	**98.5**	**98.58** ± .04
CIFAR-100							
Wide-ResNet-40-2	78.42 ± .31	-	79.4	79.3	-	79.01	**79.86** ± .19
Wide-ResNet-28-10	82.22 ± .25	83.3	82.7	82.9	83.3	82.82	**84.33** ± .17
ShakeShake-26-2x96d	83.28 ± .14	84.7	85.4	85.7	-	85.00	**86.19** ± .15
SVHN Core							
Wide-ResNet-28-10	97.12 ± .05	-	-	98.0	**98.3**	-	98.11 ± .03
SVHN							
Wide-ResNet-28-10	98.67 ± .02	98.9	98.8	98.9	**99.0**	-	98.9 ± .02
ImageNet							
ResNet-50	77.20 ± .32 (93.43 ± .11)	-	77.6 (93.7)	77.6 (**93.8**)	77.6 (**93.8**)	77.63 (-)	**78.07** ± .27 (**93.92** ± .09)

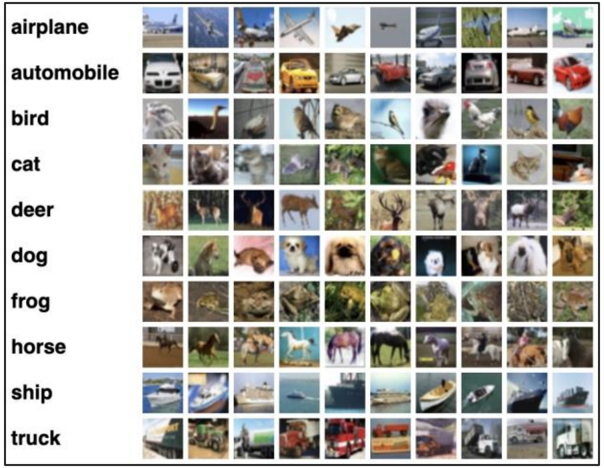
Table 2: The average test accuracies from ten runs, besides for ImageNet, where we used five runs. The 95% confidence interval is noted with ±. The trivial TA is in all benchmarks among the top-performers. The only exception is the comparison to RA's performance on the SVHN benchmarks, but this difference was non-existent in our reimplementation.

Motivation:

- Stellt eine Augmentation Library und Open Source Implementierungen der bisherigen Verfahren bereit
- Bietet eine deutlich verbesserte Datengrundlage zu Bewertung der Verfahren

Eigene Versuche & Ergebnisse

Datensatz ist nicht gleich Datensatz



CIFAR10 Datensatzausschnitt [1]

SVHN Datensatzausschnitt [2]

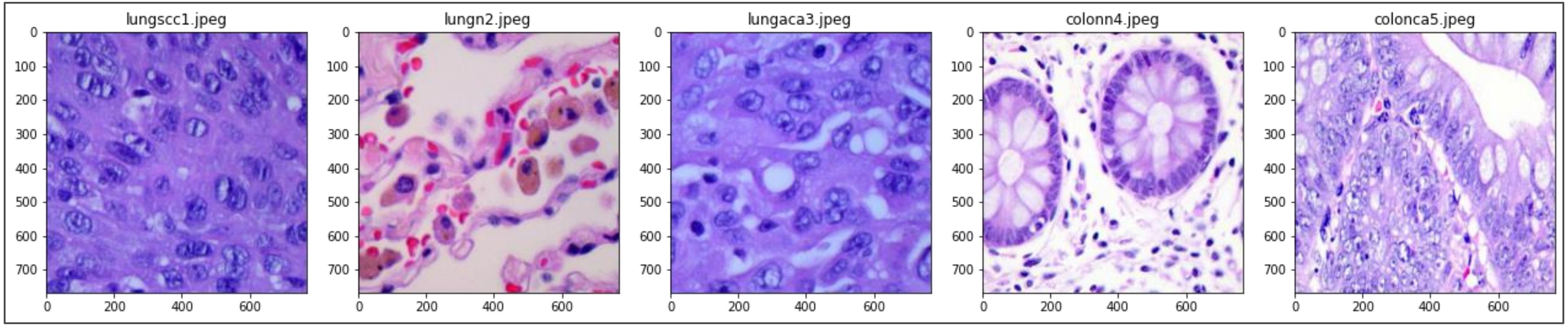
ImageNet Datensatzausschnitt [3]

[1] cs.toronto.edu/~kriz/cifar.html
[2] http://ufldl.stanford.edu/housenumbers/
[3] https://cs.stanford.edu/people/karpathy/cnnembed/

LC25000 – Informationen zum Datensatz

➢ Aufnahmen von Lungen- und Darmgewebe

- **Anzahl:** 25.000 Bilder
- **Klassen:** Multi-Class (fünf Klassen -> gutartiges Gewebe, verschiedene Krebsarten)
- **Format:** 768x768px

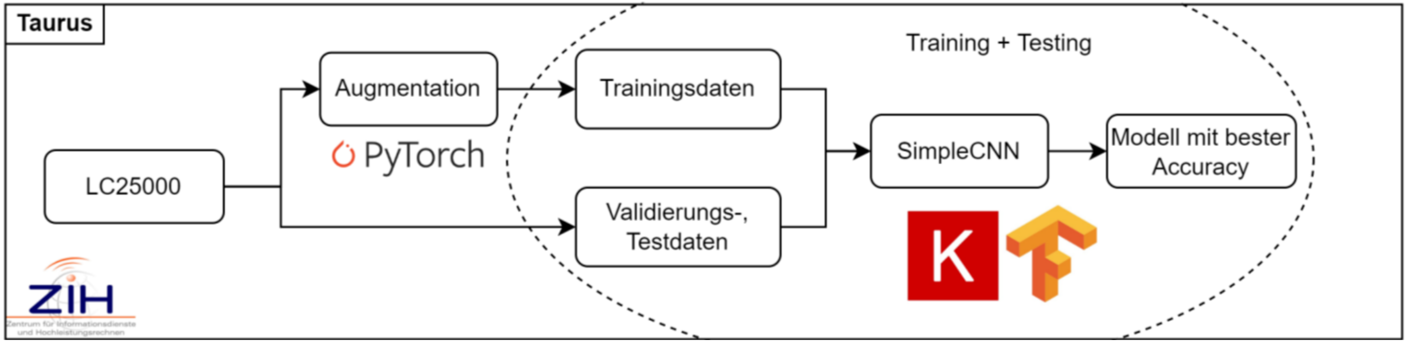


LC25000

A. A. Borkowski, M. M. Bui, L. B. Thomas, C. P. Wilson, L. A. DeLand, και S. M. Mastorides, 'Lung and Colon Cancer Histopathological Image Dataset (LC25000)'. arXiv, 2019.

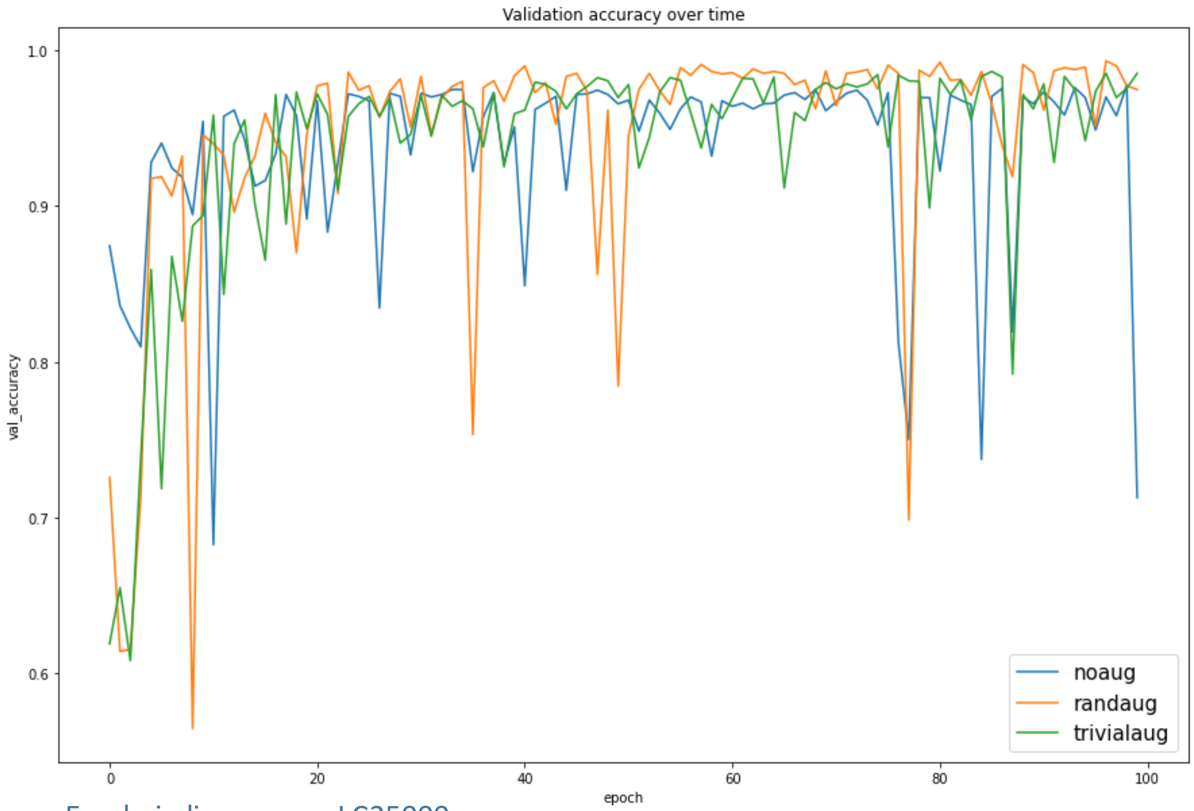
LC25000 - Versuchsaufbau

- **Datenbasis:** Skalierung auf 224x224px
- **Framework:** TensorFlow, Keras, PyTorch
- **Modell:** „SimpleCNN“ (ca. 12 Millionen trainierbare Parameter)
- **Training:** 100 Epochen, 15er Batch Size
- **Systemumgebung:** ZiH HPC-Cluster Taurus (Partition Alpha -> NVIDIA Tesla A100 40GB RAM)



Ablaufdiagramm LC25000-Training

LC25000 – Ergebnisse



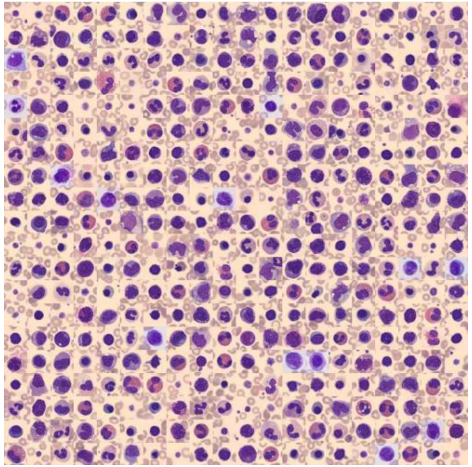
Ergebnisdiagramm – LC25000

Variante	Validation Accuracy
NoAug	97.75%
RandAug	**99.24%**
TrivialAug	98.63%

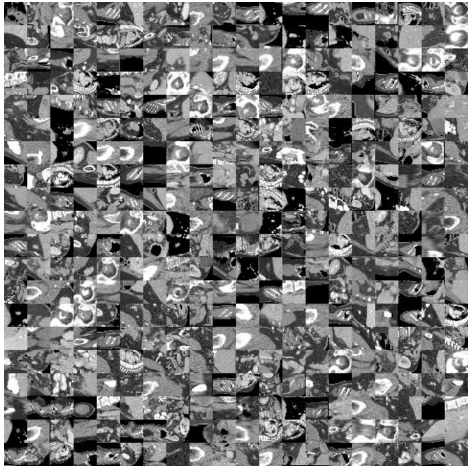
Ergebnisse – LC25000

medMNIST – Informationen zum Datensatz

- Datensatzsammlung biomedizinischer Bilder
- **Anzahl:** 18 Datensätze (2D/3D) mit 100 bis 100.000 Bilder pro Datensatz
- **Klassen:** Multi-Class, Binary-Class
- **Format:** 28x28px (2D) / 28x28x28px (3D)
- Bspw. Röntgen-, Ultraschall- und Elektronenmikroskop-Aufnahmen



BloodMNIST

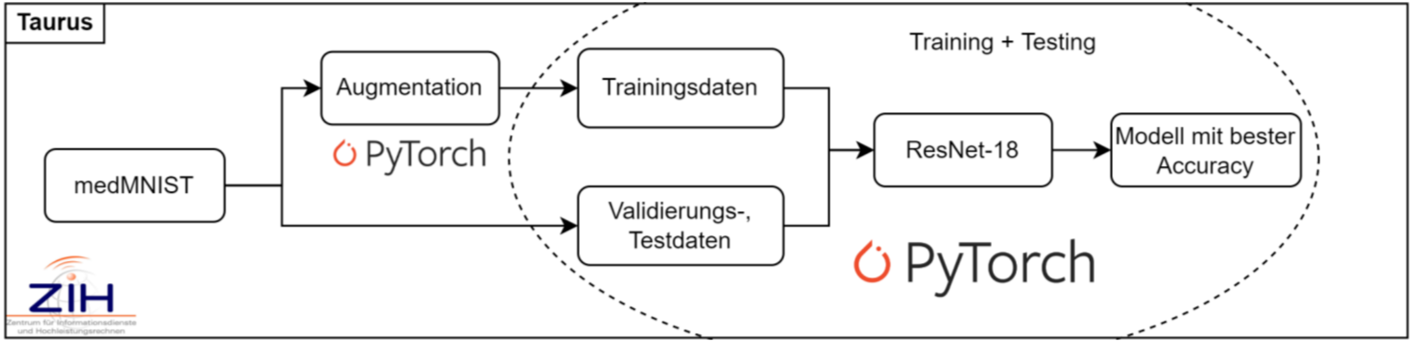


OrganMNIST3D

J. Yang κ.ά., 'MedMNIST v2: A Large-Scale Lightweight Benchmark for 2D and 3D Biomedical Image Classification'. arXiv, 2021.

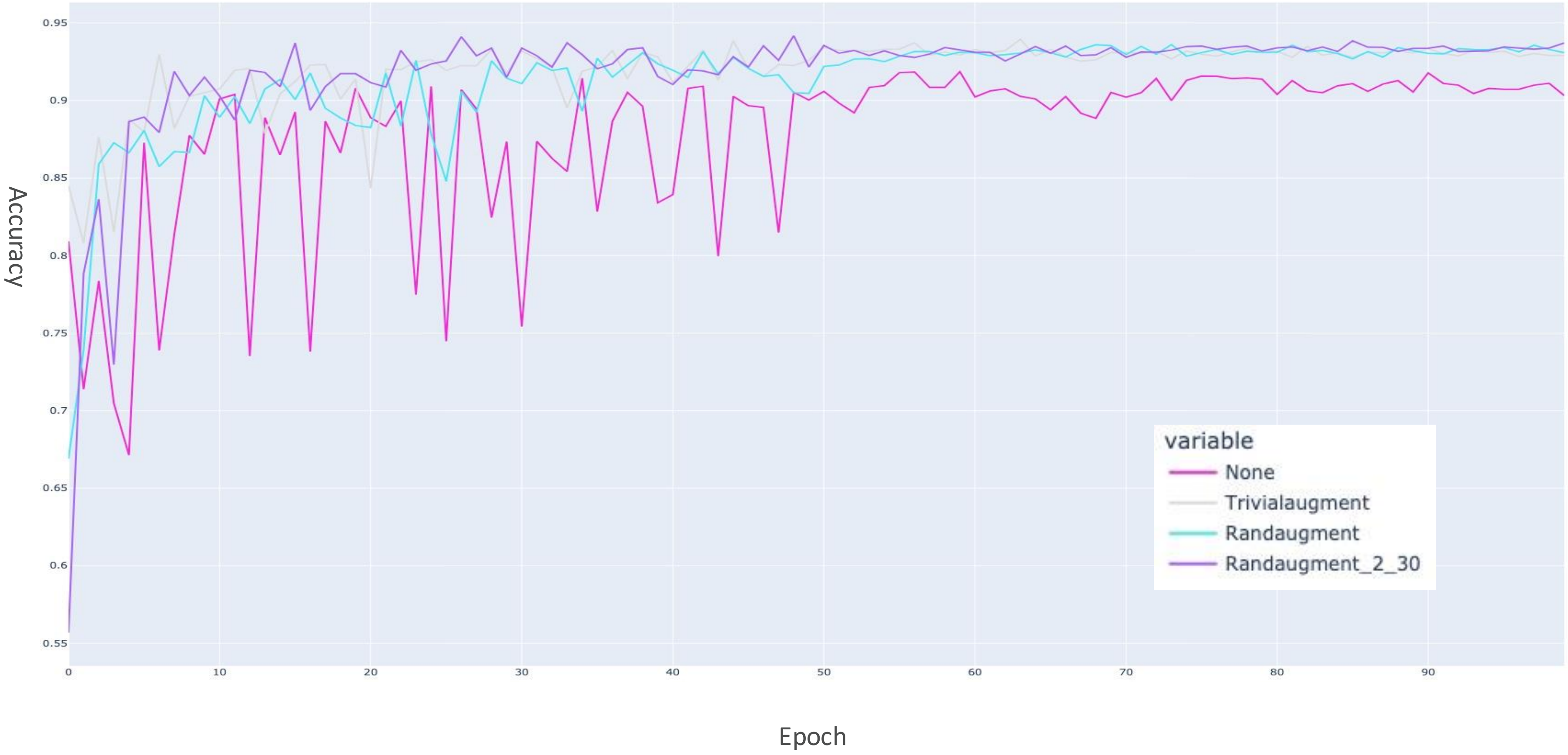
medMNIST – Versuchsaufbau

- **Datenbasis:** Alle 12 2D-Datensätze nach Klasseneinteilung
- **Framework:** PyTorch
- **Modell:** ResNet-18 (ca. 11 Millionen trainierbare Parameter)
- **Training:** 100 Epochen, 128er Batch Size
- **Systemumgebung:** ZIH HPC-Cluster Taurus (Partition Alpha -> NVIDIA Tesla A100 40GB RAM)



Ablaufdiagramm medMNIST-Training

pathMNIST – Ergebnisse



MedMNIST-Ergebnistabelle

TestACC	blood	breast	chest	derma	oct	organa	organc	organs	path	pneumonia	retina	tissue
Trivialaugment	0.96960	**0.90385**	0.93770	0.75561	0.75400	0.95877	0.93372	0.84109	0.92911	**0.91346**	0.50500	0.67187
Randaugment (TA)	0.96814	0.87179	0.93663	0.75810	0.77600	**0.96293**	0.93166	**0.84177**	0.93078	0.87660	0.53750	0.68005
Randaugment (2,30)	**0.97018**	0.82692	**0.93787**	**0.78504**	**0.78600**	0.96034	**0.93880**	0.84154	**0.93705**	0.87179	**0.56750**	**0.69126**
None	0.95586	0.89103	0.93474	0.74065	0.75100	0.93970	0.91497	0.78820	0.90320	0.85417	0.47750	0.66546

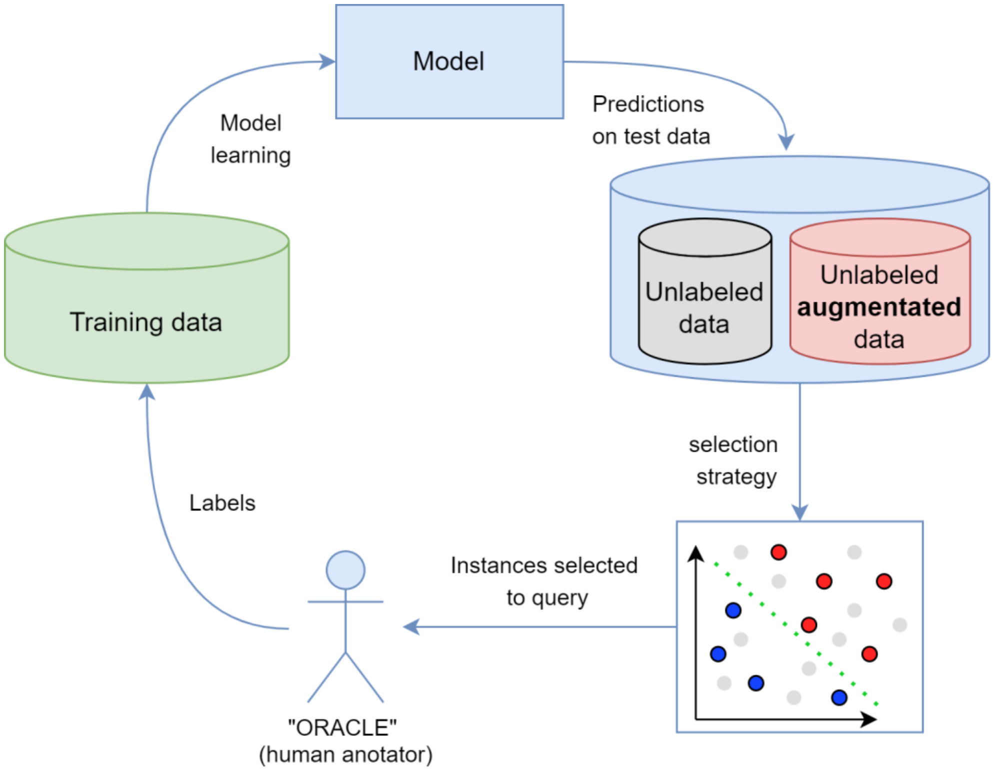
- Trivialaugment und Randaugment performen ähnlich gut,
- Beide Verfahren waren in jedem Fall besser als keine Augmentationen zu verwenden

Ausblick

Lessons Learned

- Keep it simple stupid
- Teile deinen Versuchsaufbau
- Gestalte deine Implementierung Open-Source
- Verwende TrivialAugment (für Bildklassifikation)
- Verschaffe dir einen Überblick bevor du Code schreibst

Active Learning + Data Augmentation



Erweiterter Active Learning Zyklus

Vielen Dank für die Aufmerksamkeit!

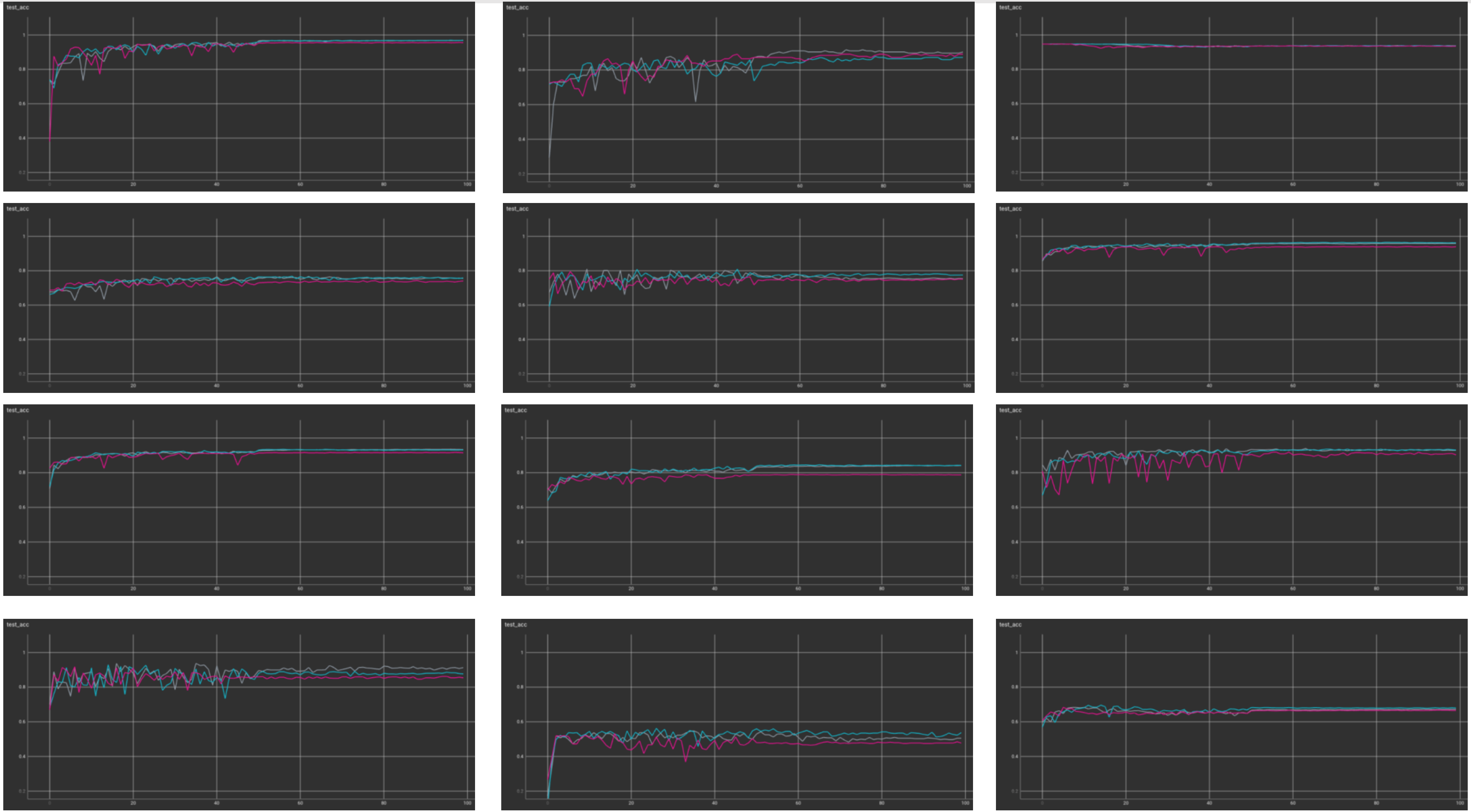
I705 Forschungs- und Entwicklungsprojekt / -seminar

Prof. Dr. Maik Thiele | Colin Simon, Serhiy Bolkun, Kevin Kirsten

GitHub

Unsere Ergebnisse unter:

https://github.com/ColinS97/AL4ML


test_acc
TrivialAugment
RandAugment
No Augment