

Seminar Text Analytics

# Image Understanding

**Lernen visueller Konzepte bei  
Inferieren ohne fine-tuning**

**Goethe-Universität Frankfurt  
Kenan Khauto**

# Agenda

1. Einführung
2. Analyse von CLIP
3. Kontrastives Lernen im Detail
4. Anwendungen
5. Herausforderungen und Grenzen
6. Mögliche Verbesserungen
7. Literatur

# Einführung

## Die sich wandelnde Landschaft der Bilderkennung

- Exponentielles Wachstum digitaler Bilder in sozialen Medien, Überwachungssystemen, im Gesundheitswesen und bei autonomen Fahrzeugen.
- Traditionelle Bilderkennungsverfahren basieren auf umfangreichen, manuell beschrifteten Datensätzen.
- Zeitaufwendige und kostspielige Erstellung dieser Datensätze.



Quelle: (generiert von ChatGPT4)

# Einführung

## Grenzen der traditionellen Ansätze

- Notwendigkeit großer Mengen an beschrifteten Daten begrenzt die Skalierbarkeit.
- Schwierigkeiten bei der Generalisierung auf neue, unbekannte Kategorien ohne umfangreiches Neutraining.

# Einführung

## Einführung in das Zero-Shot-Learning

- Zero-Shot-Learning ermöglicht die Erkennung von Objekten/Bildern ohne vorherige direkte Trainingsbeispiele.
- Nutzung semantischer Informationen und Beziehungen zwischen Konzepten für die Inferenz.
- Potenzial für signifikante Verbesserungen in der Effizienz und Flexibilität der Bilderkennung.

# Einführung

## CLIP: Die Verbindung von Vision und Sprache

- CLIP (Contrastive Language–Image Pre-training) nutzt natürliche Sprachbeschreibungen, um visuelle Konzepte zu lernen.
- Ermöglicht das Verständnis und die Durchführung einer Vielzahl visueller Aufgaben ohne spezifisches Training für jede Aufgabe.
- Beispiele für die Vielseitigkeit und Effizienz von CLIP in unterschiedlichen Anwendungen.

# Einführung

## Seminarfrage

Wie können KI-Modelle wie CLIP visuelle Konzepte effektiv durch Inferenz verstehen und interpretieren, ohne dass ein umfangreiches Fine-Tuning erforderlich ist ?

# Analyse von CLIP

## Was ist CLIP? (Radford u. a. 2021)

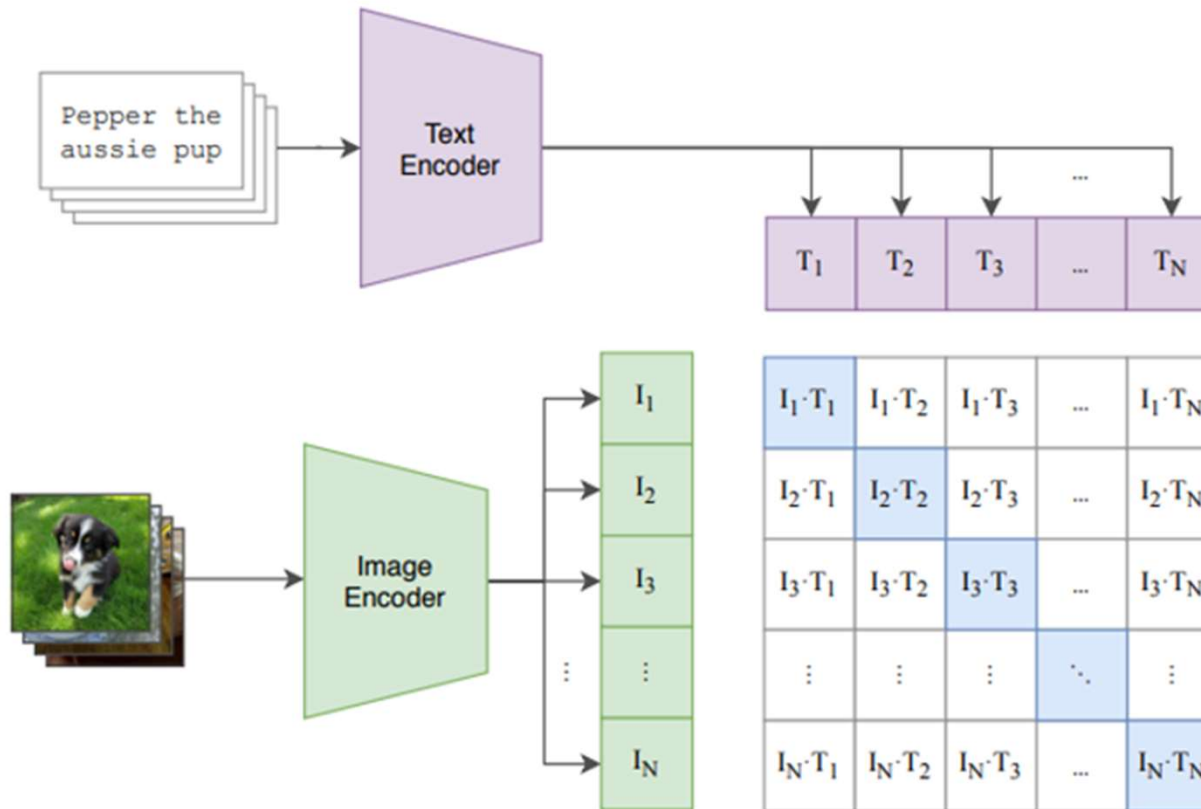
- CLIP ist ein neuronales Netzwerk, das anhand einer Vielzahl von Bildern und den zugehörigen Textbeschreibungen trainiert wurde.
- Dieses Training ermöglicht es ihm, sowohl visuelle als auch textuelle Eingaben zu verstehen und zu interpretieren.



# Analyse von CLIP

## Architektur (Radford u. a. 2021, S 2)

(1) Contrastive pre-training

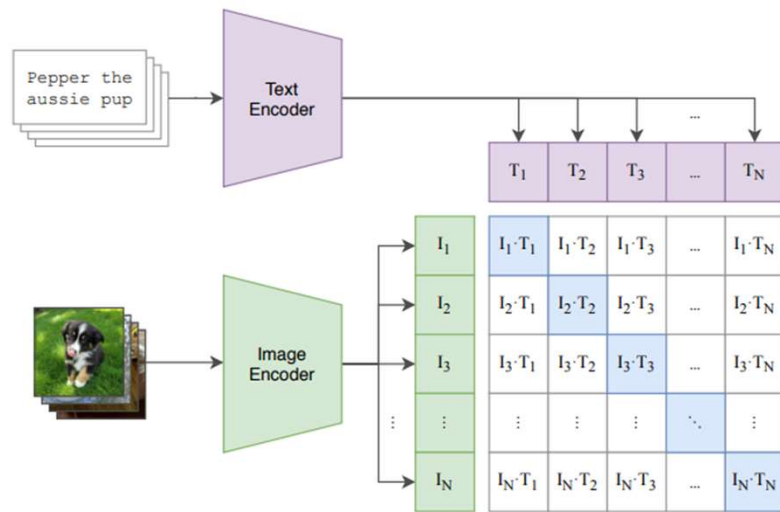


CLIP, Bild von (Radford u. a. 2021, S 2)

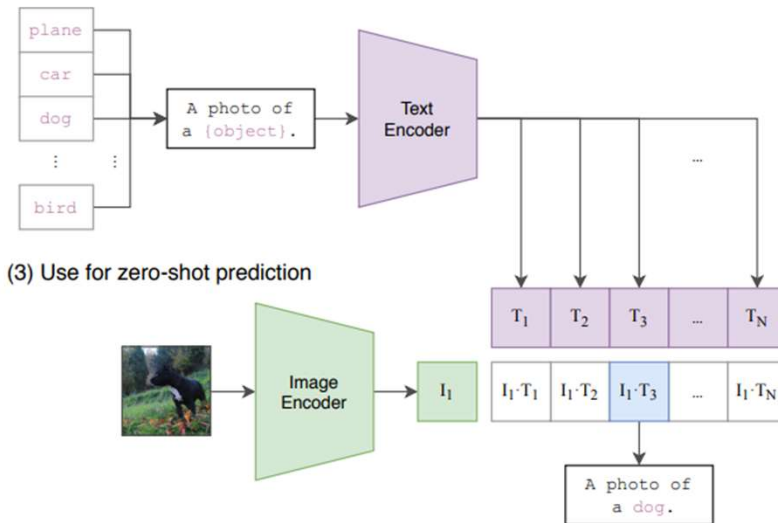
# Analyse von CLIP

## Architektur (Radford u. a. 2021, S 2)

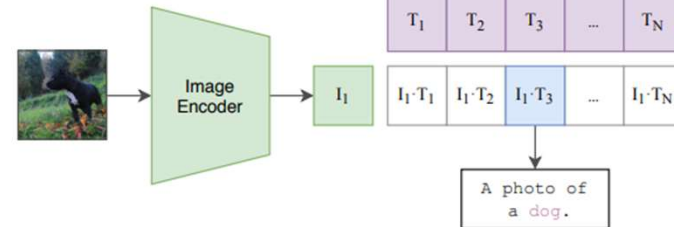
(1) Contrastive pre-training



(2) Create dataset classifier from label text



(3) Use for zero-shot prediction



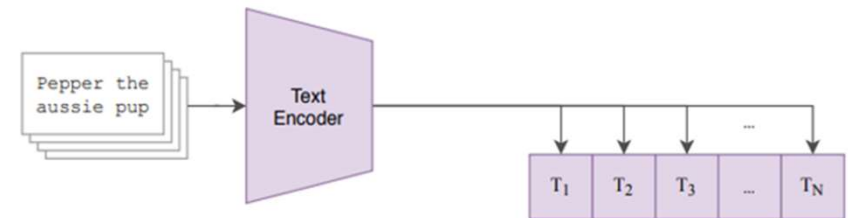
CLIP, Bild von (Radford u. a. 2021, S 2)

# Analyse von CLIP

## Text-Encoder (Vaswani u. a. 2017)

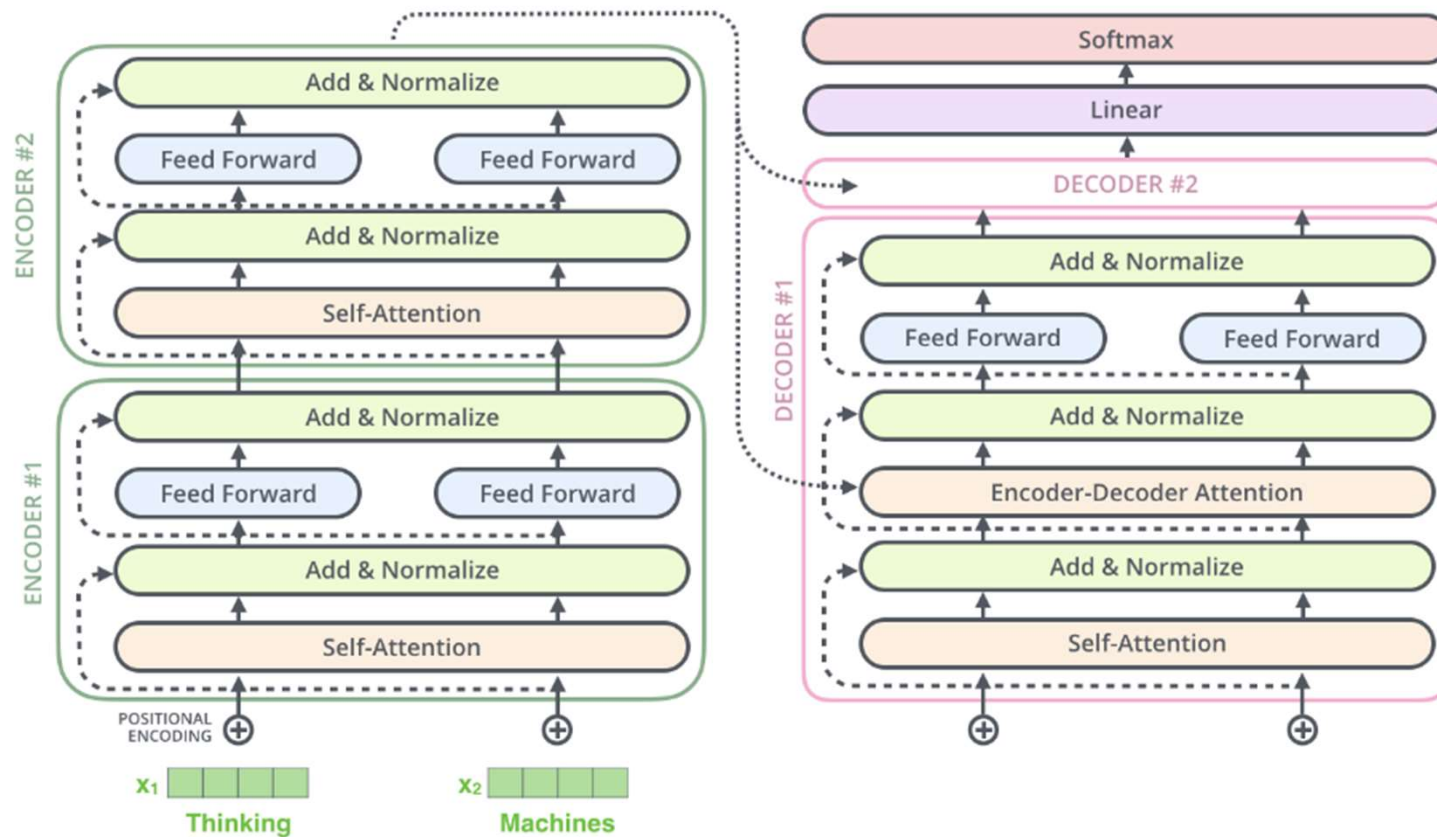
- Der Text-Encoder in CLIP basiert auf einer Transformer-Architektur.
- Jedes Wort im Text wird in eine Vektordarstellung umgewandelt.
- Mit self-attention wird die Bedeutung jedes Wortes im Kontext des gesamten Text verstanden.
- **Ziel:** eine Darstellung des Textes zu erzeugen, die dessen semantischen Inhalt in Bezug auf mögliche Bildinhalte widerspiegelt.

(1) Contrastive pre-training



# Analyse von CLIP

## Text-Encoder (Vaswani u. a. 2017)



Transformer mit 2 Encoders, 2 Decoders und FCL,  
<https://jalammar.github.io/illustrated-transformer/>

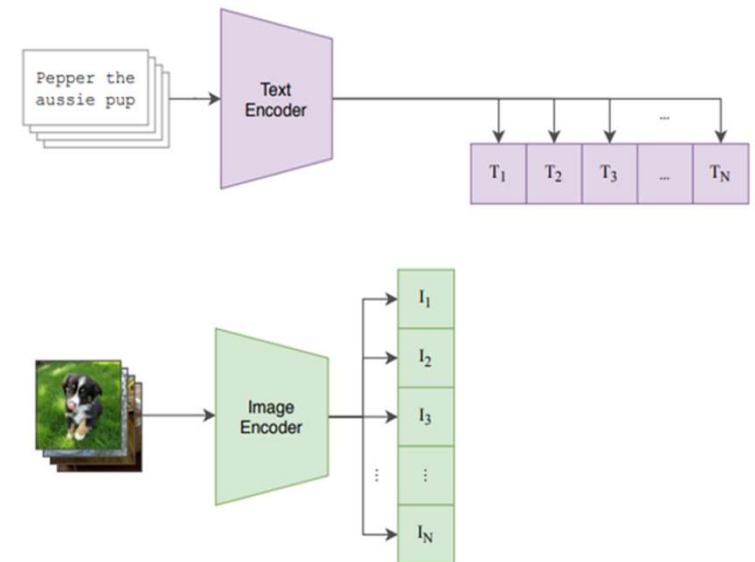
# Analyse von CLIP

## Bild-Encoder (O'Shea und Nash 2015)

Der Bild-Encoder ist nicht ausschließlich auf CNN beschränkt, sondern kann auch Vision Transformers umfassen.

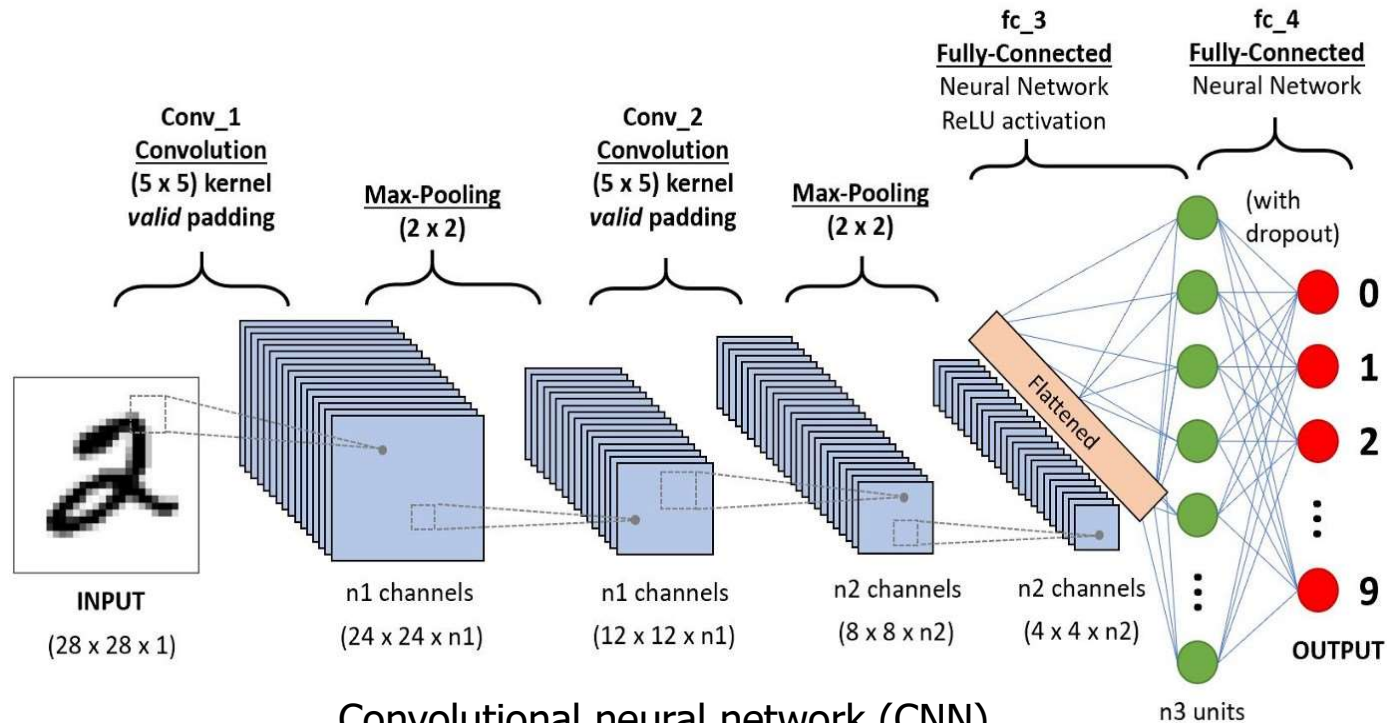
- CNN-basierte Encoder:
  - Traditionelle CNN-Architekturen
  - Effektiv in der Erkennung lokaler Muster
  - Wandeln das Bild in eine Vektorrepräsentation um, die die visuelle Inhalte des Bildes kodieren

(1) Contrastive pre-training



# Analyse von CLIP

## Bild-Encoder (O'Shea und Nash 2015)



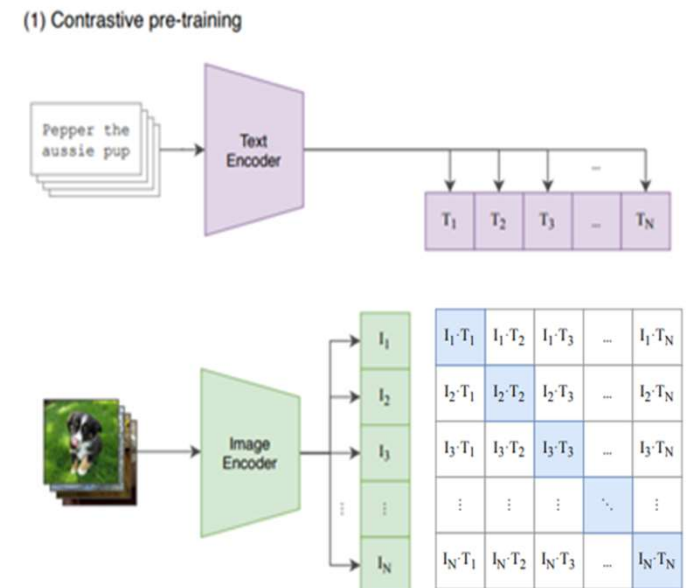
Convolutional neural network (CNN),  
<https://paperswithcode.com/methods/category/convolutional-neural-networks>

# Analyse von CLIP

## Bild-Encoder (Dosovitskiy u. a. 2020)

Der Bild-Encoder ist nicht ausschließlich auf CNN beschränkt, sondern kann auch Vision Transformers umfassen.

- Vision Transformers:
  - Zerlegen das Bild in eine Sequenz von Patches
  - Die Patches werden ähnlich wie Wörter in einem Satz behandelt
  - Effektiv in der Erkennung sowohl lokale als auch globale Kontextinformationen aus dem Bild



CLIP, Bild von (Radford u. a. 2021, S 2)



# Analyse von CLIP

## Integration und gemeinsamer Einbettungsraum (Radford u. a. 2021)

Die zentrale Innovation von CLIP besteht darin, dass beide Encoder – der Text- und der Bild Encoder – darauf trainiert sind, ihre Ausgaben in einem **gemeinsamen Einbettungsraum** zu repräsentieren.

- Korrespondierende Paare nah beieinander
- Kontrastives Lernen: Minimierung der Distanz zwischen übereinstimmenden Bild-Text-Paaren und Maximierung der Distanz zwischen nicht übereinstimmenden Paaren









Quelle: (generiert von ChatGPT4)



# Analyse von CLIP

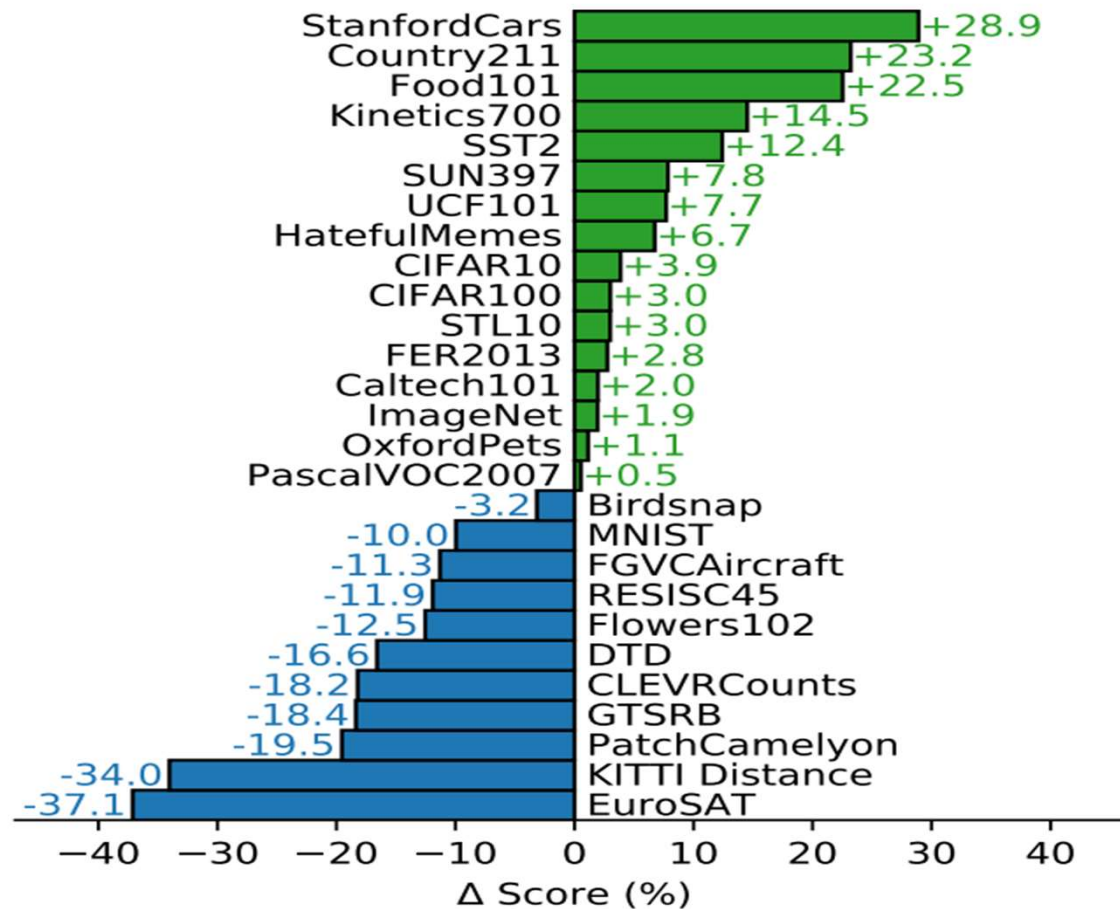
## Vergleichsanalyse (Radford u. a. 2021, S 15)

	Dataset Examples	ImageNet ResNet101	Zero-Shot CLIP	$\Delta$ Score
ImageNet		76.2	76.2	0%
ImageNetV2		64.3	70.1	+5.8%
ImageNet-R		37.7	88.9	+51.2%
ObjectNet		32.6	72.3	+39.7%
ImageNet Sketch		25.2	60.2	+35.0%
ImageNet-A		2.7	77.1	+74.4%

Resnet-101 fine-tuned vs. zero-shot CLIP, Bild von (Radford u. a. 2021, S 15)

# Analyse von CLIP

## Vergleichsanalyse (Radford u. a. 2021, S 8)



Linear Probe ResNet50 vs. zero-shot CLIP, Bild von (Radford u. a. 2021, S 8)

# Kontrastives Lernen im Detail

## Vektoreinbettungen (Radford u. a. 2021, S 5)

Sei  $I$  ein Bild und  $T$  ein Text, dann

Bild-Einbettung:  $v_I = f_{Bild}(I)$

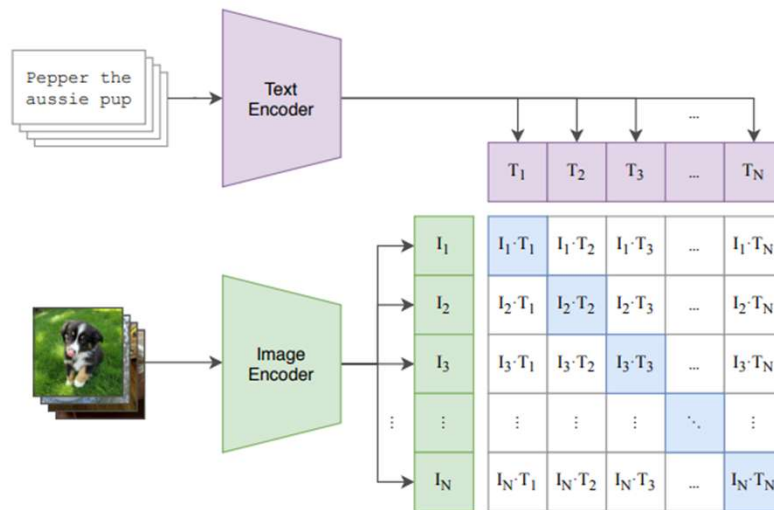
Text-Einbettung:  $v_T = f_{Text}(T)$

Wobei  $f_{Bild}(I)$  und  $f_{Text}(T)$  die Funktionen des Bild- bzw. Text-Encoders sind.

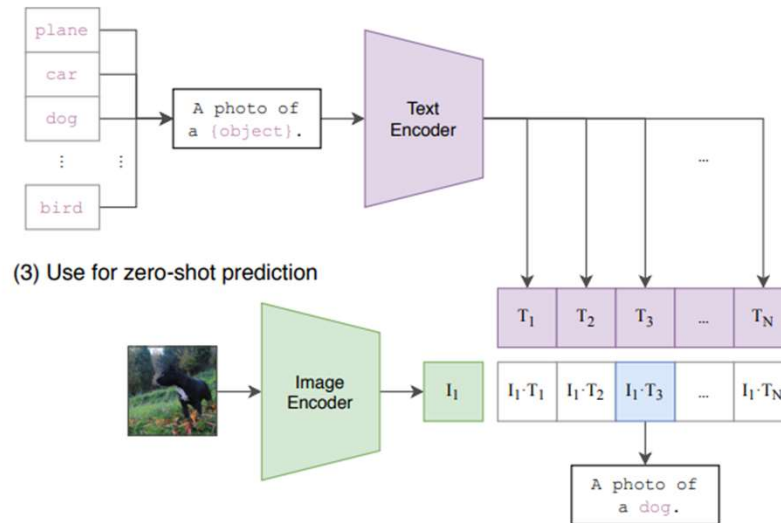
# Kontrastives Lernen im Detail

## Ähnlichkeitsberechnung (Radford u. a. 2021, S 2)

(1) Contrastive pre-training



(2) Create dataset classifier from label text



(3) Use for zero-shot prediction

CLIP, Bild von (Radford u. a. 2021, S 2)

# Kontrastives Lernen im Detail

## Ähnlichkeitsberechnung (Radford u. a. 2021, S 5)

Die Ähnlichkeit zwischen einem Bild- und einem Textvektor wird durch das Skalarprodukt ihrer normalisierten Vektoren berechnet:

$$\text{sim}(\mathbf{v}_I, \mathbf{v}_T) = \frac{\mathbf{v}_I \cdot \mathbf{v}_T}{\|\mathbf{v}_I\| \cdot \|\mathbf{v}_T\|}$$

Dann wird die Verlust für ein Paar  $(\mathbf{I}, \mathbf{T})$  berechnet als:

$$L(\mathbf{I}, \mathbf{T}) = -\log \frac{\exp\left(\frac{\text{sim}(\mathbf{v}_I, \mathbf{v}_T)}{\tau}\right)}{\sum_{T'} \exp\left(\frac{\text{sim}(\mathbf{v}_I, \mathbf{v}_{T'})}{\tau}\right)} - \log \frac{\exp\left(\frac{\text{sim}(\mathbf{v}_T, \mathbf{v}_I)}{\tau}\right)}{\sum_{I'} \exp\left(\frac{\text{sim}(\mathbf{v}_T, \mathbf{v}_{I'})}{\tau}\right)}$$

- $\tau$  ist ein Temperatur-Parameter
- Die Summen laufen über alle Texte und Bilder jeweils

# Kontrastives Lernen im Detail

## Gesamtverlust (Radford u. a. 2021, S 5)

Für ein Batch mit  $N$  Bild-Text Paare, wird die Gesamtverlust mit:

$$L_{total} = \frac{1}{N} \sum_{i=1}^N L(I_i, T_i)$$

berechnet.

Die symmetrische Natur der Loss-Funktion (symmetric Cross Entropy Loss) stellt sicher, dass die Bild-zu-Text- und Text-zu-Bild-Vorhersagen während des Trainings gleichmäßig betont werden

# Kontrastives Lernen im Detail

## Numpy\_like Pseudocode (Radford u. a. 2021, S 5)

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t            - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```



# Anwendungen

## Anwendungen

- **Zero-Shot-Lernen:** CLIP kann Bilder klassifizieren, in dem es die Ähnlichkeiten zwischen Bildern und Textbeschreibungen nutzt.
- **Bild- und Textsuche:** Es ermöglicht die Suche nach Bildern mit textuellen Beschreibungen und umgekehrt, die Suche nach Texten, die zu einem Bild passen.
- **Content Moderation:** CLIP kann zur automatischen Erkennung und Filterung unangemessener Inhalte in Bildern und Texten eingesetzt werden.
- **Automatische Bildbeschriftung:** Generierung von Textbeschreibungen für Bilder.
- **Sentiment-Analysis in Bildern:** Erkennung der Stimmung oder des Gefühls, das ein Bild vermittelt.
- Und noch mehr ...



# Anwendungen

## Beispielcode für Bild-Klassifizierung (Pinecone 2023)

```
from tqdm.auto import tqdm

preds = []
batch_size = 32

for i in tqdm(range(0, len(imagenette), batch_size)):
    i_end = min(i + batch_size, len(imagenette))
    images = processor(
        text=None,
        images=imagenette[i:i_end]['image'],
        return_tensors='pt'
    )['pixel_values'].to(device)
    img_emb = model.get_image_features(images)
    img_emb = img_emb.detach().cpu().numpy()
    scores = np.dot(img_emb, label_emb.T)
    preds.extend(np.argmax(scores, axis=1))
```

Der komplette Code findet man auf  
<https://github.com/KenanKhauto/zero-shot-learning>

```
true_preds = []
for i, label in enumerate(imagenette['label']):
    if label == preds[i]:
        true_preds.append(1)
    else:
        true_preds.append(0)

sum(true_preds) / len(true_preds)
```

✓ 0.0s

0.9831847133757962

# Anwendungen

## Beispielcode für Bild-Klassifizierung

Zero-Shot CLIP auf CIFAR10 mit verschiedenen Prompts,  
<https://github.com/KenanKhauto/zero-shot-learning>

Prompt	Accuracy	Comment
'This is a {}.'	0.696	not so good
'A {}'	0.672	Poor result
'{}'	0.708	Could be better...
'This image contains {}'	71.5	Improvement, but not significantly better
'There is {}'	71.5	Improvement, but not significantly better
'a photo of a {}'	0.716	better
'you can see a {}'	0.716	better
'A {} is in the picture'	0.726	better
'In this image you can see a {}'	0.727	Best result so far
'You can see a {} in this image'	0.727	Best result so far
'An {} is waiting'	0.733	best

# Grenzen

## Grenzen

- **Verzerrungen und Vorurteile:** Wie viele KI-Modelle kann auch CLIP-Verzerrungen aufweisen, die in den Trainingsdaten vorhanden sind. Dies kann zu unfairen oder voreingenommenen Ergebnissen führen, besonders bei der Analyse von Bildern und Texten aus verschiedenen Kulturen und sozialen Gruppen.
- **Abhängigkeit von der Textqualität:** Die Leistung von CLIP ist stark abhängig von der Qualität und Relevanz der Textbeschreibungen. Unpräzise oder irreführende Texte können zu fehlerhaften Ergebnissen führen.
- **Generalisierungsfähigkeit:** Obwohl CLIP gut in der Lage ist, Konzepte zu generalisieren, kann es Schwierigkeiten geben, sehr spezifische oder seltene Objekte und Szenarien korrekt zu erkennen und zuzuordnen.
- **Komplexität und Ressourcenanforderungen:** CLIP-Modelle sind groß und rechenintensiv, was ihre Anwendbarkeit in ressourcenbeschränkten Umgebungen einschränkt.

# Mögliche Verbesserungen

## Verbesserungen

- **Diversifizierung der Trainingsdaten:** Um Verzerrungen und Vorurteile zu reduzieren, sollten die Trainingsdaten vielfältiger und repräsentativer gestaltet werden. Dies schließt Daten aus verschiedenen Kulturen, Sprachen und sozialen Gruppen ein.
- **Erweiterte Kontextanalyse:** Die Integration zusätzlicher Kontextinformationen kann helfen, die Genauigkeit der Bild-Text-Zuordnungen zu verbessern, besonders bei komplexen oder mehrdeutigen Szenen.
- **Interdisziplinäre Ansätze:** Zusammenarbeit mit Experten aus verschiedenen Bereichen wie Sozialwissenschaften, Ethik und Kunst, um die Anwendungen und Implikationen von CLIP besser zu verstehen und zu steuern.

# Quellen

- Dosovitskiy, Alexey u. a. (2020). „An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale“. In: *arXiv preprint arXiv:2010.11929*.
- O'Shea, Keiron und Ryan Nash (2015). „An Introduction to Convolutional Neural Networks“. In: *arXiv preprint arXiv:1511.08458*.
- Pinecone (2022). *Zero Shot Object Detection with OpenAI's CLIP*. Accessed on: 2024-01-22. URL: <https://www.pinecone.io/learn/series/image-search/zero-shot-object-detection-clip/>.
- (2023). *Zero-shot Image Classification with OpenAI's CLIP*. Accessed on: 2024-01-22. URL: <https://www.pinecone.io/learn/series/image-search/zero-shot-image-classification-clip/>.
- Radford, Alec u. a. (2021). *Learning Transferable Visual Models From Natural Language Supervision*. arXiv: [2103.00020](https://arxiv.org/abs/2103.00020) [cs.CV].
- Vaswani, Ashish u. a. (2017). „Attention is all you need“. In: *Advances in neural information processing systems*, S. 5998–6008.

# Fragen ?