TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

AUSTRIAN INSTITUTE OF TECHNOLOGY

Deep Learning for Video Analysis – Guest Lecture

LVA Video Analysis (188.329)
Winter Semester 2022
Andreas Kriegler

Modalities

- Contacts:
  - andreas.kriegler@tuwien.ac.at
  - margrit.gelautz@tuwien.ac.at

- Slides will be available in the TUWEL course

- Machine learning (ML) for Computer Vision builds on decades of mathematical frameworks – here it is packed into ~15 mins
- Some of you will not have acquired the prerequisite knowledge yet, so you may struggle to understand everything as we go – that‘s okay ☺

If you want to help in keeping the number of errors at a reasonable level send any that you find to andreas.kriegler@tuwien.ac.at

Literature

- ML & DL use applied statistics, linear algebra & calculus:
  - Mathematical foundations and many common algorithms of machine learning – the ML “bible”: [1]
  - The application of deep learning in neural networks: [2], [3]
  - Artificial intelligence in general and multi-agent theory: [4]
- Lectures:
  - The problems of high dimensional learning – lecture 2 of the AMMI 2021 GDL100 course: [5]
  - TU Wien - 194.100 Theoretical Foundations and Research Topics in Machine Learning [6]
  - Stanford - CS229 Machine Learning [7]
  - Stanford - CS231n Convolutional Neural Networks for Visual Recognition [8]
  - MIT - Deep Learning and Artificial Intelligence Lectures [9]
- Videos:
  - Deep Learning Series, 3Blue1Brown [10]
  - Mathematics for Machine Learning, Ulrike von Luxburg [11]

Outline

I. Introduction to ML & DL (~15 mins.)

II. Video lecture (~40 mins.)

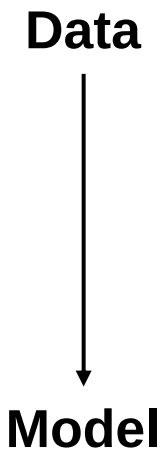
III. Discussion (~15 mins.)

IV. Current state-of-the-art (~20 mins.)

Machine learning basics

- In classical programming we define rules for input → output relations

- In machine learning (ML) we use data to
  - 1. Generate our model (**learning**)
  - 2. Apply it to new observations (**inference**/prediction)



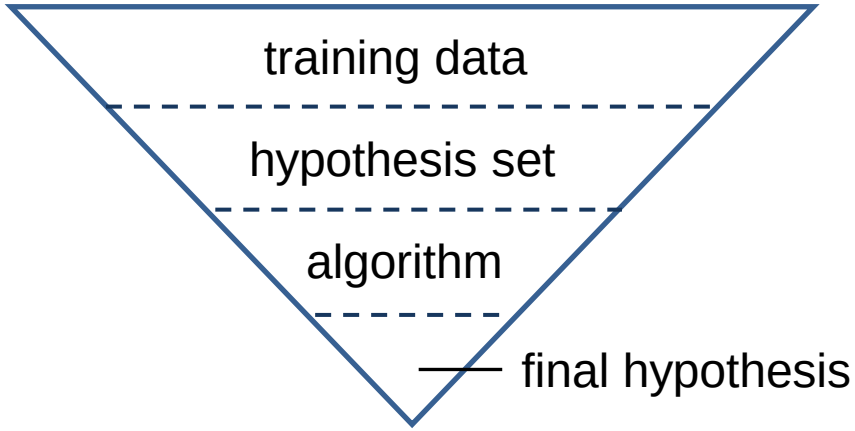


Figure recreated from [12]

ML basics – supervised classification

ML example transcribed from [1]

- We have: a **training set** of N observations of x, $\boldsymbol{x} := (x_1, \dots, x_N)^\mathsf{T}$ with corresponding **target** observations t, $\boldsymbol{t} := (t_1, \dots, t_N)^\mathsf{T}$
- We want: predict $\hat{t}$ for new $\hat{x}$, using parameters $\boldsymbol{\theta}$
- **Regression** if $\boldsymbol{t} \subset \mathbb{R}$ or n-way **classification** if $\boldsymbol{t}$ is categorical $t \in \{0, \dots, n-1\}$
- We can, for example, try to fit a polynomial curve

$$f = y(x, \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_M x^M = \sum_{j=0}^{M} \theta_j x^j \qquad (1)$$

- We can calculate any loss/error function, often L1 or L2

$$L1(\boldsymbol{\theta}) = \frac{1}{2} \sum_{n=1}^{N} \{|y(x_n, \boldsymbol{\theta}) - t_n|\} = ||\boldsymbol{\theta}||_1 \qquad (2)$$

$$L2(\boldsymbol{\theta}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \boldsymbol{\theta}) - t_n\}^2 = ||\boldsymbol{\theta}||_2 \qquad (3)$$

ML basics – supervised classification

ML example transcribed from [1]

$$L(\boldsymbol{\theta}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{\theta}) - t_n\}^2 \quad (4)$$

- This will **overfit**, i.e. perform well on training but poorly on test samples
- To prevent this we add a term (λ) for **regularization**

$$\tilde{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{\theta}) - t_n\}^2 + \frac{\lambda}{2}||\boldsymbol{\theta}||^2 \quad (5)$$

- Now we can update – **optimize** – our parameters $\boldsymbol{\theta}$ with e.g. curve fitting methods

This was a simple example to introduce common terminology - in typical ML problems we have a (non-)convex loss function to use **gradient descent** techniques [13]

In ML the hat $\hat{x}$ is typically used to denote targets and the snake $\tilde{x}$ for estimates of some variable x

Deep learning – artificial neural networks

- Deep learning with multilayer perceptrons (MLP) since 1965 [14]
- Dealing with vanishing gradients since 1991 [15, 16], deep since 2012 [17]
- Now models with up to 10^{12} parameters trained on cloud-based tensor or graphical processing unit (TPU/GPU) clusters [18]

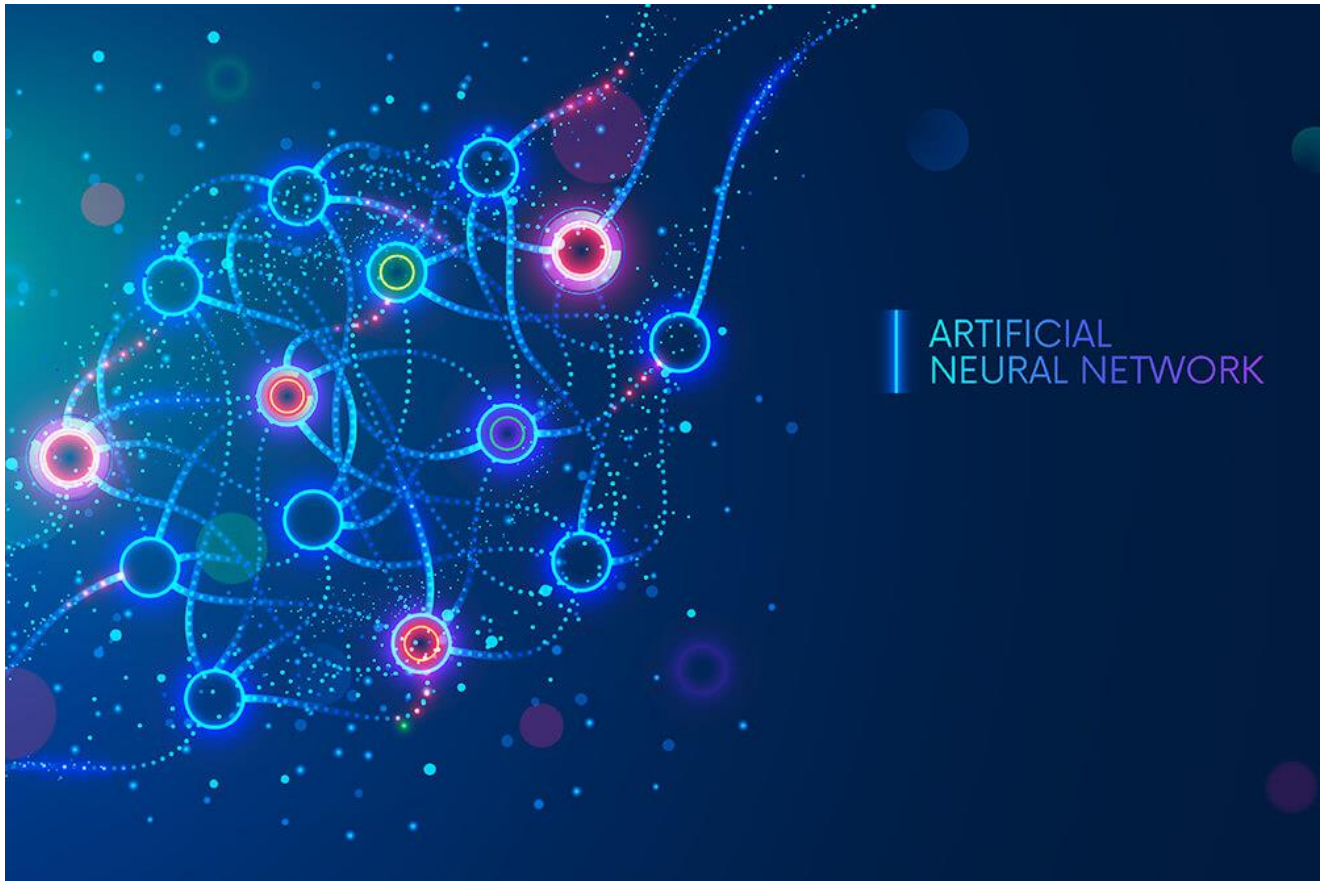


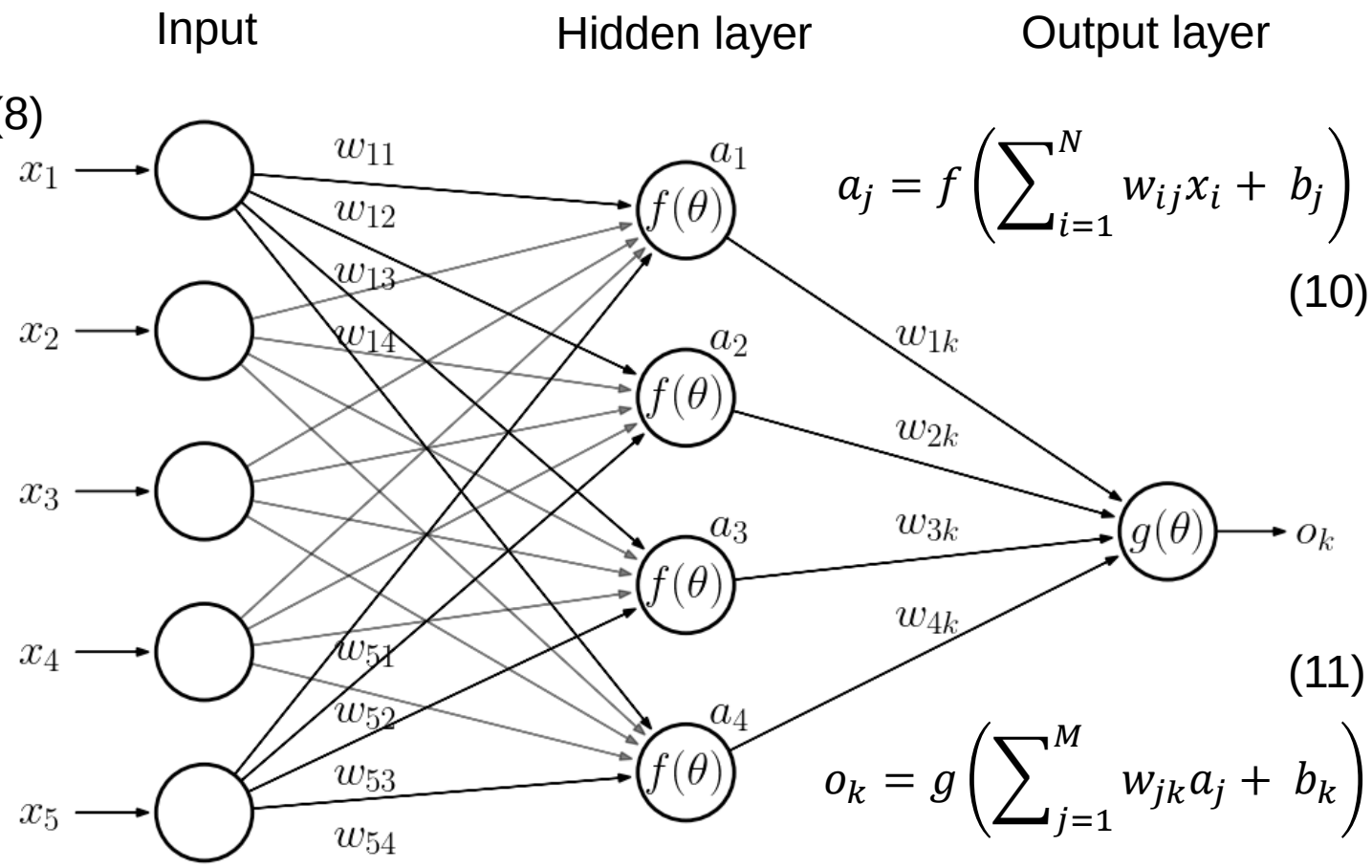
Figure taken from [19]

Deep learning – 1-hidden layer neural network

- **Activations** f in hidden neurons
  - Sigmoid (old): $\phi(z) = \frac{1}{1+e^{-z}}$ (8)
  - **ReLU** (rectified linear unit): $\mathrm{ReLU}(z) = \max(0, z)$ (9)
- **Output** g **often softmax** σ

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- **Weights** w scale the function and **bias** b shifts it
- N, M: number of input neurons – here 5 and 4



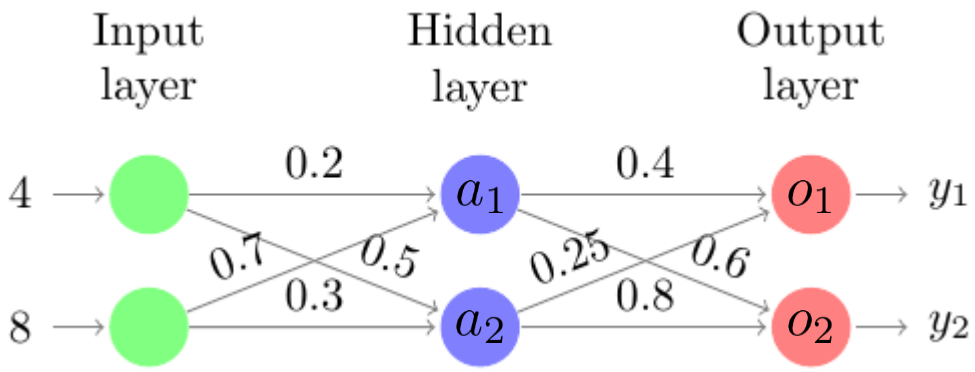
$$a_j = f\left(\sum_{i=1}^{N} w_{ij}x_i + b_j\right) \quad (10)$$

$$o_k = g\left(\sum_{j=1}^{M} w_{jk}a_j + b_k\right) \quad (11)$$

Figure taken from [20]

NN forward-pass example

- Given the input, weights, active bias neurons in the hidden and output layer and standard sigmoid(x) activation function and softmax(x) squashing, what is the output y_1 and y_2



- Prediction would be the label associated to y_2

$$a_1 = sig(\sum_{i=1}^{2}(w_{i1}x_i) + 1) \approx 0.838$$

$$a_2 = sig(\sum_{i=1}^{2}(w_{i2}x_i) + 1) \approx 0.808$$

$$o_1 = \sum_{j=1}^{2}(w_{j1}a_j) + 1 \approx 1.537$$

$$o_2 = \sum_{j=1}^{2}(w_{j2}a_j) + 1 = 2.15$$

$$y_1 = softmax(o_1) = \frac{e^{o_1}}{\sum_{k=1}^{2} e^{o_k}} = 0.351$$

$$y_2 = softmax(o_2) = \frac{e^{o_2}}{\sum_{k=1}^{2} e^{o_k}} = 0.649$$

Training NNs – obtaining the loss

- The output vector of the NN is squashed to predictions $\hat{y} \in (0, 1]$ using **softmax**

$$\hat{y} = \sigma(\boldsymbol{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{6}$$

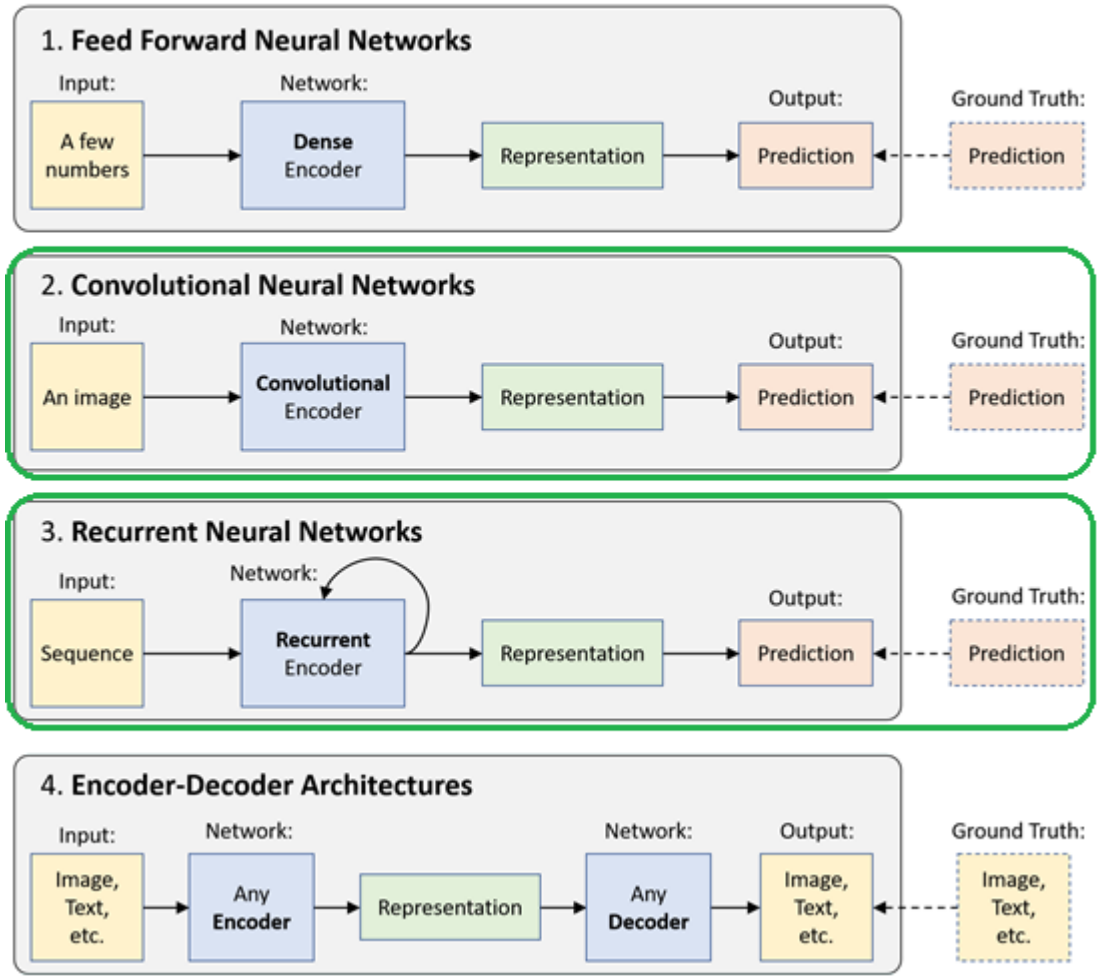
- This „distribution over categorials“ allows probabilistic reasoning [74]
- We apply **cross-entropy loss** (simplified version) with $\hat{\boldsymbol{y}}$ and ground truth (gt) $\boldsymbol{y}$

$$L = -\boldsymbol{y} \cdot \log(\hat{\boldsymbol{y}}) \tag{7}$$

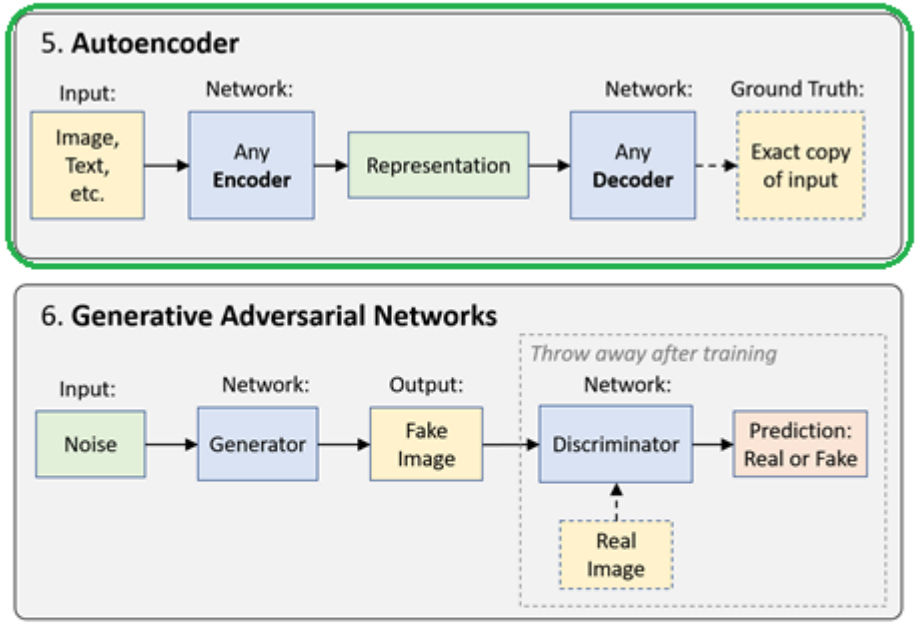
- NNs are typically updated using a variation of gradient descent: stochastic gradient descent (**SGD**)

Learning families and neural networks

Supervised Learning



Unsupervised Learning



Reinforcement Learning

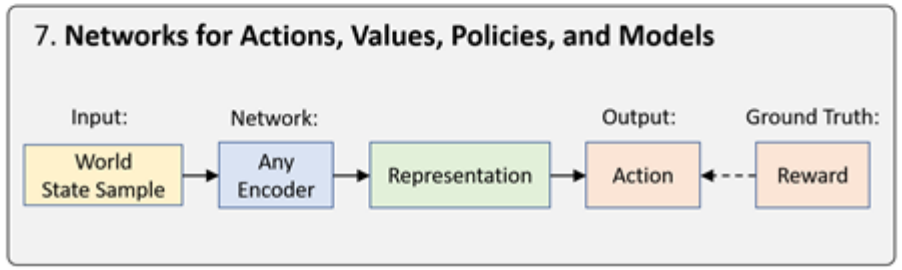


Figure slightly modified and taken from [9]

Supervised Convolutional Neural Networks (CNNs)

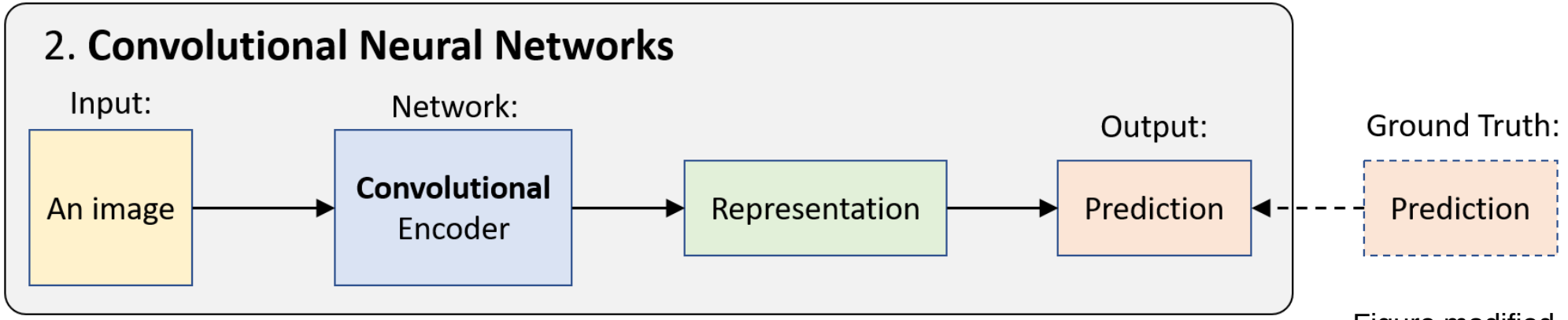
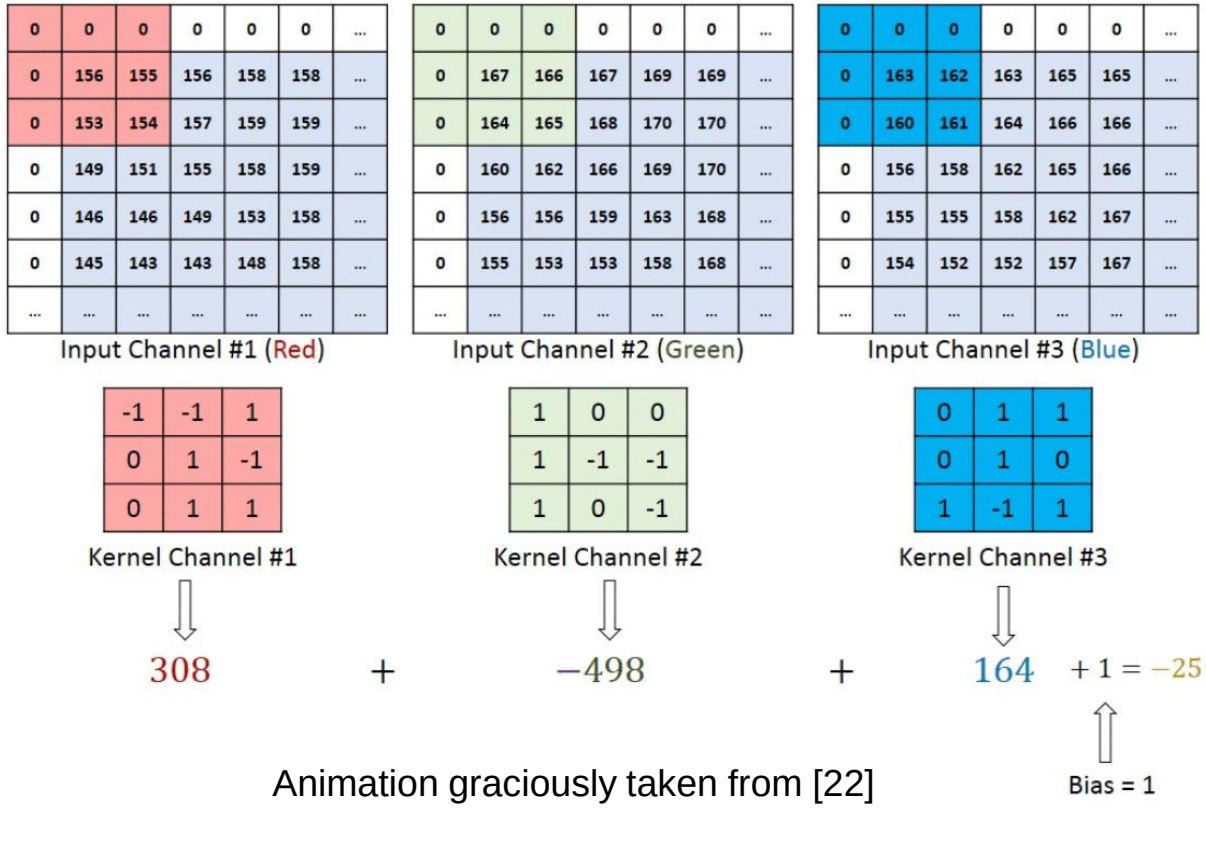


Figure modified and taken from [9]

1. We operate on images, particularly the **RGB** channels
2. The **convolutional layers** act as **encoders** for feature representations
3. From those representations we obtain predictions
4. We compare predictions with the **ground truth (gt)** via a loss function
5. Backpropagate the loss using gradient descent
6. Update our weights, i.e. **kernel entries**

Convolution operation in CNNs

- Invented in 1979 [21] – see [8] for the Stanford course
- We learn kernel entries using backpropagation



Animation graciously taken from [22]

Discrete convolution (cross-correlation)

$$(f \star h)[n] = \sum_{m=-M}^{M} f[n-m]h[m] \quad (12)$$

Easily differentiable[?]

$$\frac{\partial}{\partial x}(h \star f) = \left(\frac{\partial}{\partial x} h\right) \quad f \quad (13)$$

★is typically used for the convolution (sum-of-products) operation

Discrete convolution example

- Given below input patch and the kernel, what is the result of a discrete convolution?
- What would this filter do to an entire image?

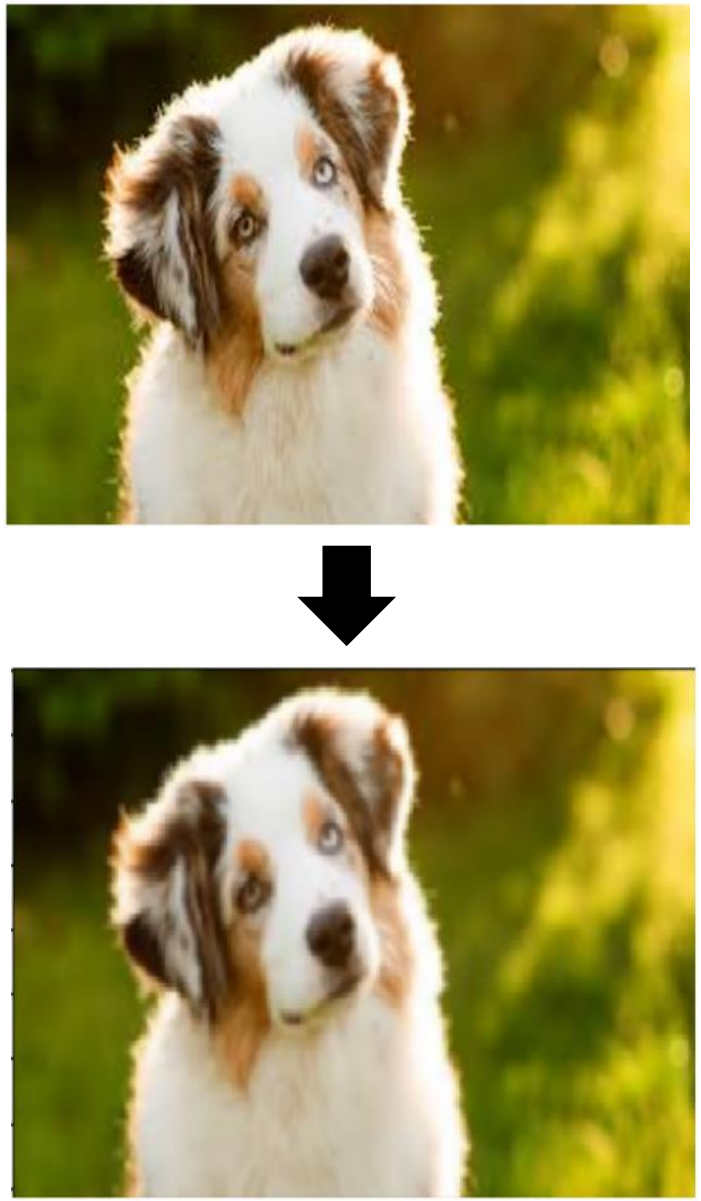
128	190	125
170	130	140
187	220	150

Input

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Kernel

- 1/9 * (128 + 190 + 125 + 170 + 130 + 140 + 187 + 220 + 150) = **160**
- It would **blur** an image, since it takes the average



Images taken from [65]

Typical CNN architecture

- Convolutional layers, activations, pooling etc. → feature-maps
- Fully connected (FC) layer followed by softmax
- Obtain a score (0, 1] at every neuron → maximum-a-posteriori rule for classification
- Differentiable cross-entropy loss → we can use gradient descent

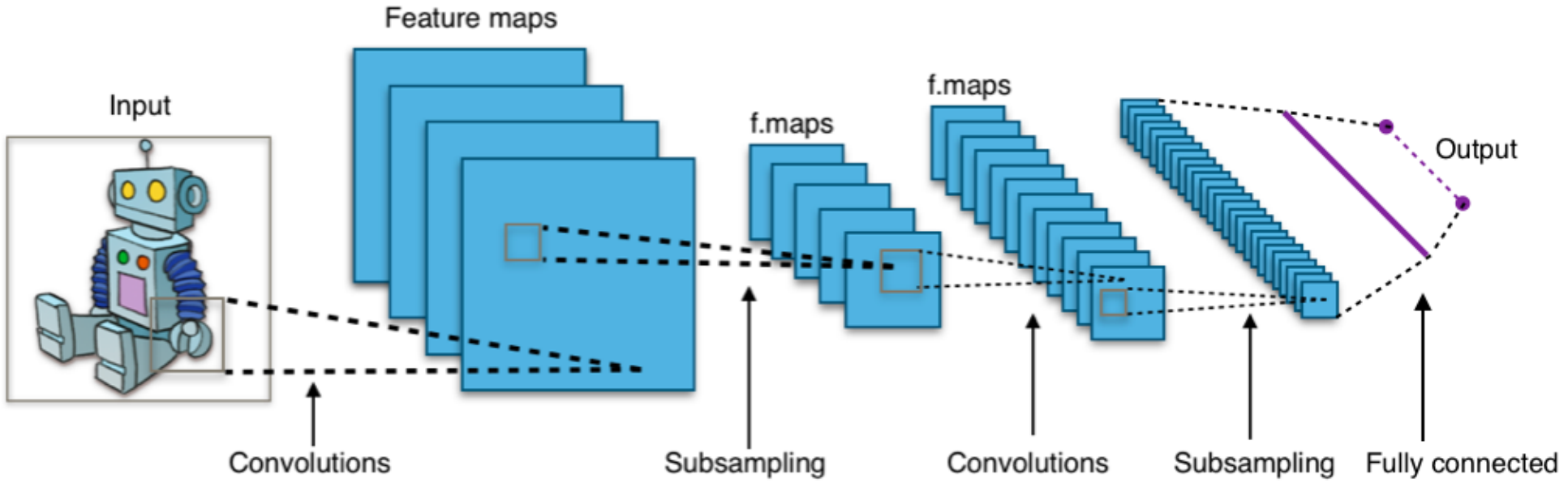


Figure taken from [23]

CNNs as powerful feature extractors

- Augmented RGB images
- The features become increasingly complex
- Visualizations on the right [24] are „projected activations of selected feature maps“
- No heatmaps but highlights of contributing features

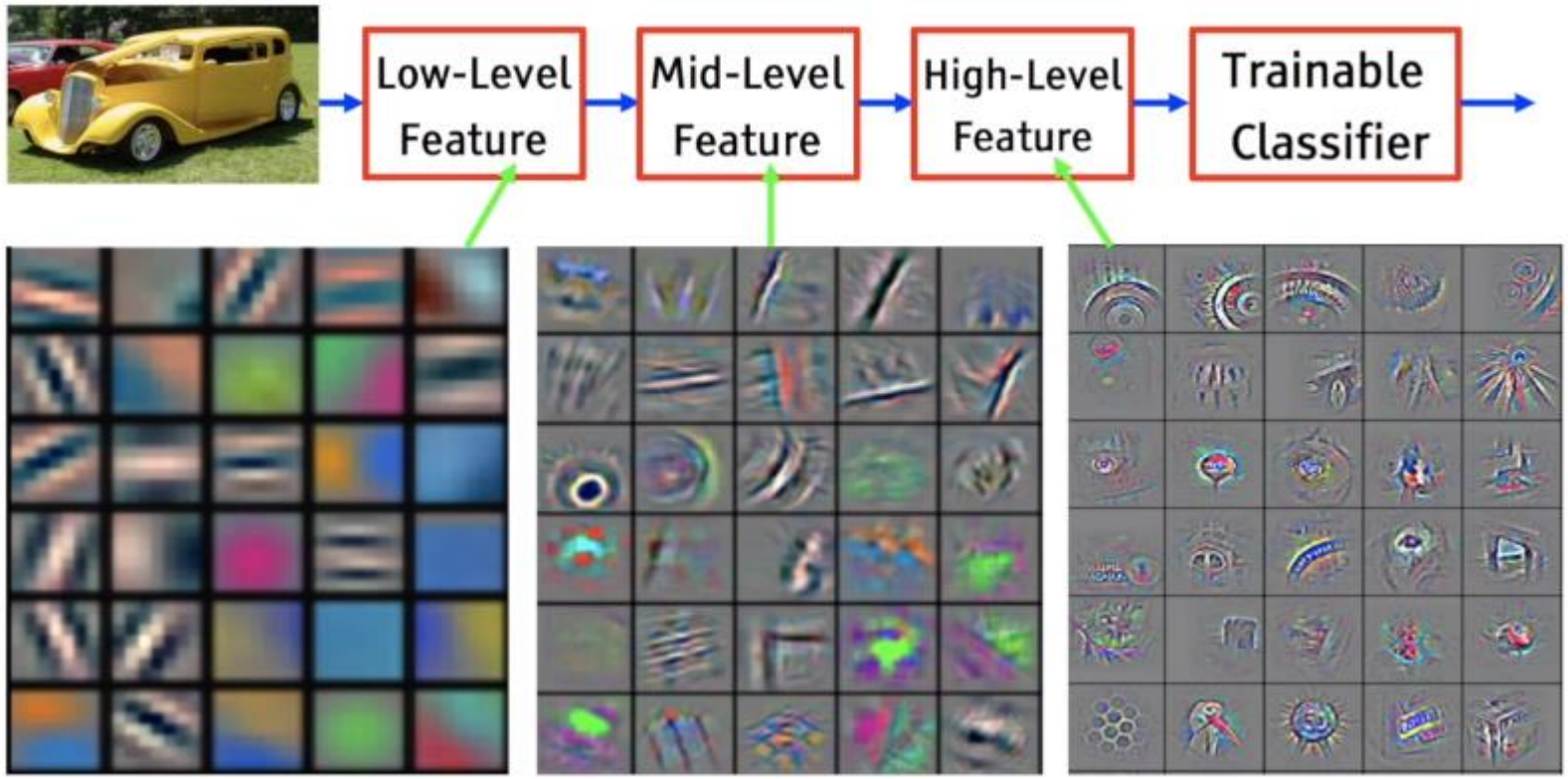


Figure taken from [24]

Recurrent neural networks (RNNs)

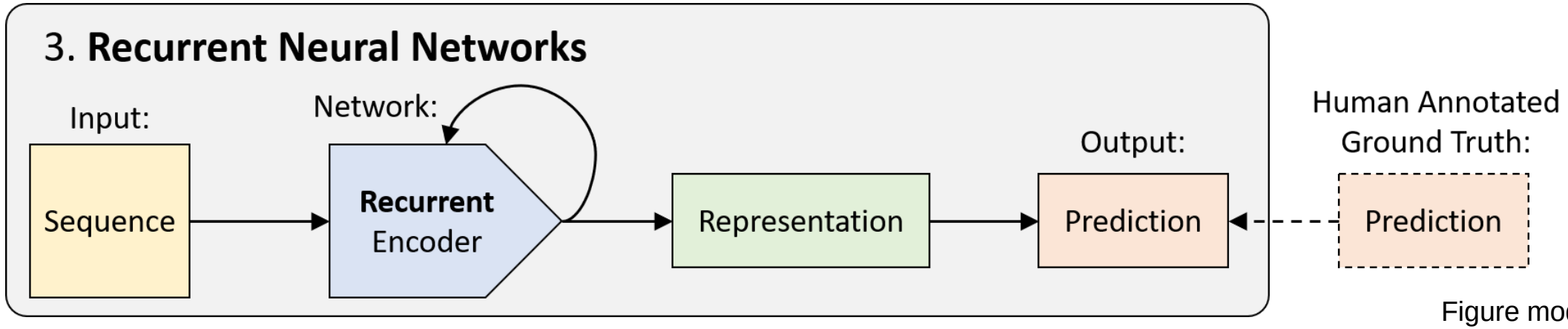


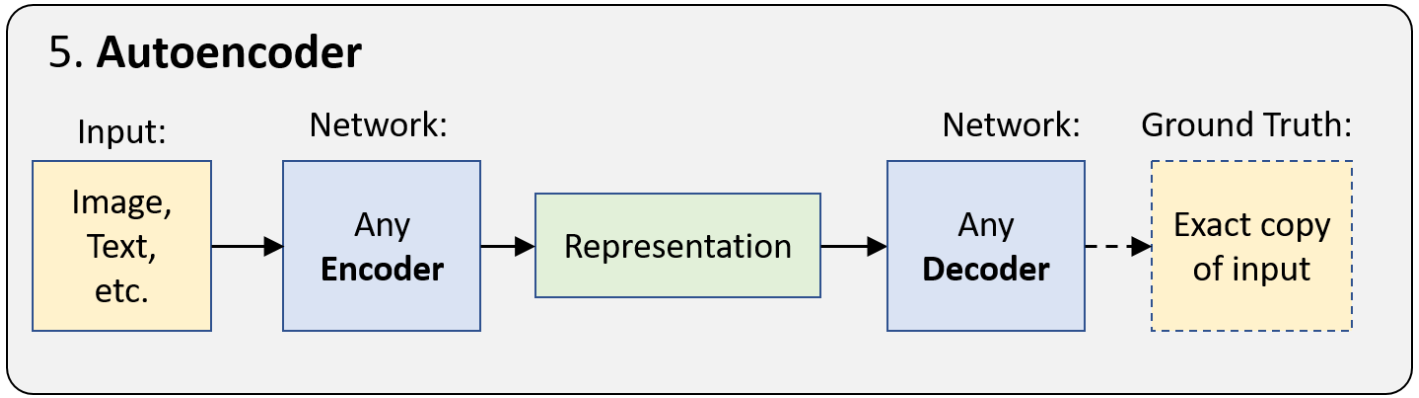
Figure modified and taken from [9]

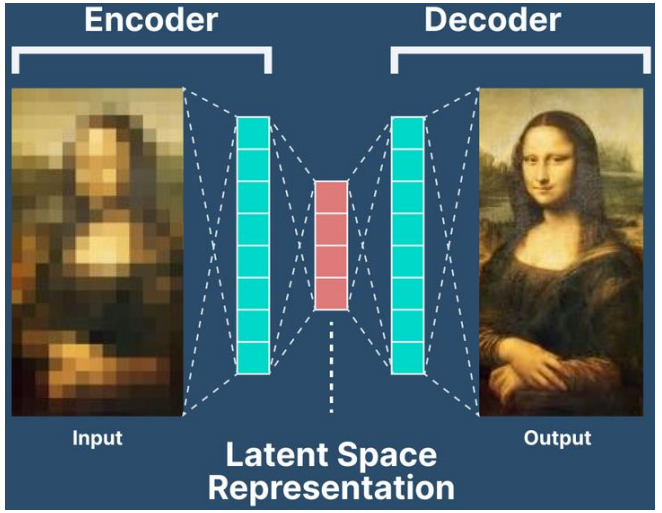
1. Work on temporal **sequences** of data (images, pointclouds, text etc.)
2. Train in cycles with a **state memory** – share weights across time
3. Obtain weights at any time-step (unroll) for a regular feed forward network
4. **Long short-term memory (LSTM)** [16] and Gated Recurrent Units [25][1]
5. Applications largely in **Natural Language Processing** and Modeling
6. LSTMs used by the „Big Five“ in billions of mobile devices [26]

[1] For a comparison between the two see [27] or an explanation in simple terms see [30]

Autoencoder

→ Figure modified and taken from [31]
↓ Figure modified and taken from [9]





1. **Encoder-Decoder** architectures use encoded, learned representations to decode to a high-dimensional output (not prediction) that can be of different modality
2. Autoencoders are **unsupervised** models trying to **recreate the input exactly**
3. A highly meaningful yet **low-dimensional representation** is learned
4. Its ground truth is the input, so **no human annotation** is required
5. Applications use these unsupervised embeddings in **representation learning**

Deep Learning Video Analysis

- Summer School Deep Learning for Computer Vision, UPC Barcelona 2018

https://www.youtube.com/watch?v=dY7j5dBqS5g [32]

Video lecture discussion questions

1. What are some general benefits/drawbacks of using videos vs. images?
2. How can self-supervision from videos deliver ground-truth estimates?
3. Explain the four presented architectures for sequence processing (strengths/drawbacks)
4. Explain differences in shallow vs. deep feature map activations under redundancy
5. How is redundancy used to speed up single-frame models?
6. What fundamental proposition was made that vastly limits the scope for applications?

Discussion

1. What are general benefits/drawbacks of using videos vs. images?

Temporal coherence enables learning entirely new things as well as an understanding closer to human perception [32] while the redundancy in the signals of the image channels is a necessary liability significantly increasing computational demands.

2. How can self-supervision from videos deliver ground-truth estimates?

The order of frames in sequences can be seen as a proxy ground-truth label for obtaining models in an unsupervised setting with feature representations following temporal constraints [33]. Motion in videos generate flow fields and assuming a stationary observer this can serve as a proxy for foreground-background segmentation [34]. Finally in videos with audio channels the representations of visual and auditory content should correspond enabling prediction of one using the other [35].

Discussion

3. Explain the four presented architectures for sequence processing in own words

- **Singe frame models** work on individual frames and use a combination (pooling) for video classification (e.g. [36]) but temporal order is ignored
- **CNN + RNN** combination uses RNN‘s to build awareness of the temporal order of the CNN features (e.g. [37]) but the RNN steps must be taken sequentially
- **3DCNNs** to process chunks/clips of ~16 frames (e.g. [38]), using pretrained 2D CNNs filters that can be replicated (and scaled) for „low-pass filtering“ (e.g. [39, 40]) but arbitrary assumptions are made and many „wheels“ to tune exist
- **Two-stream CNNs** a single-frame model for RGB and a stream for optical flow on sequenced images (e.g. [41]) but fails with ego-motion and scales poorly

Discussion

4. Explain differences in shallow vs. deep feature map activations under redundancy

Shallow layers, i.e. layers closer to the input, hold low-level features more related to patterns of colour changes that are explainably via short-distance relations such as object boundaries, corners, occlusion etc. Deep layers, with a much larger receptive field and more pooling/averaging operations resemble a representation that more closely mirrors semantic concepts that do not change as quickly (see [42]). This effect is easily observed when CNNs are applied to camera-static video inputs.

5. How is redundancy used to speed up single-frame models?

Process multiple images consecutively in the CNN at different depths, related by time steps (e.g. [43]). This again builds on the assumption that signals in videos do not change drastically from one frame to the next (i.e. unedited/uncut videos). Or use an RNN to learn what frames can be skipped (e.g. [44]).

Discussion

6. What fundamental proposition was made that vastly limits the scope for applications?

It is constrained to be a simple image classification or even video classification task!

This is like exclusively describing the Lord of the Rings trilogy as „fantasy“ :-(

- We will move on to the SotA – for video classification and more comprehensive learning tasks

MoViNets [45] – mobile & fast video recognition

1. Employ NAS (neural architecture search) to generate efficient architecture reducing 3D CNN computation cost

2. Use a stream buffer to decouple memory from video clip length

3. Ensembling technique to improve accuracy further

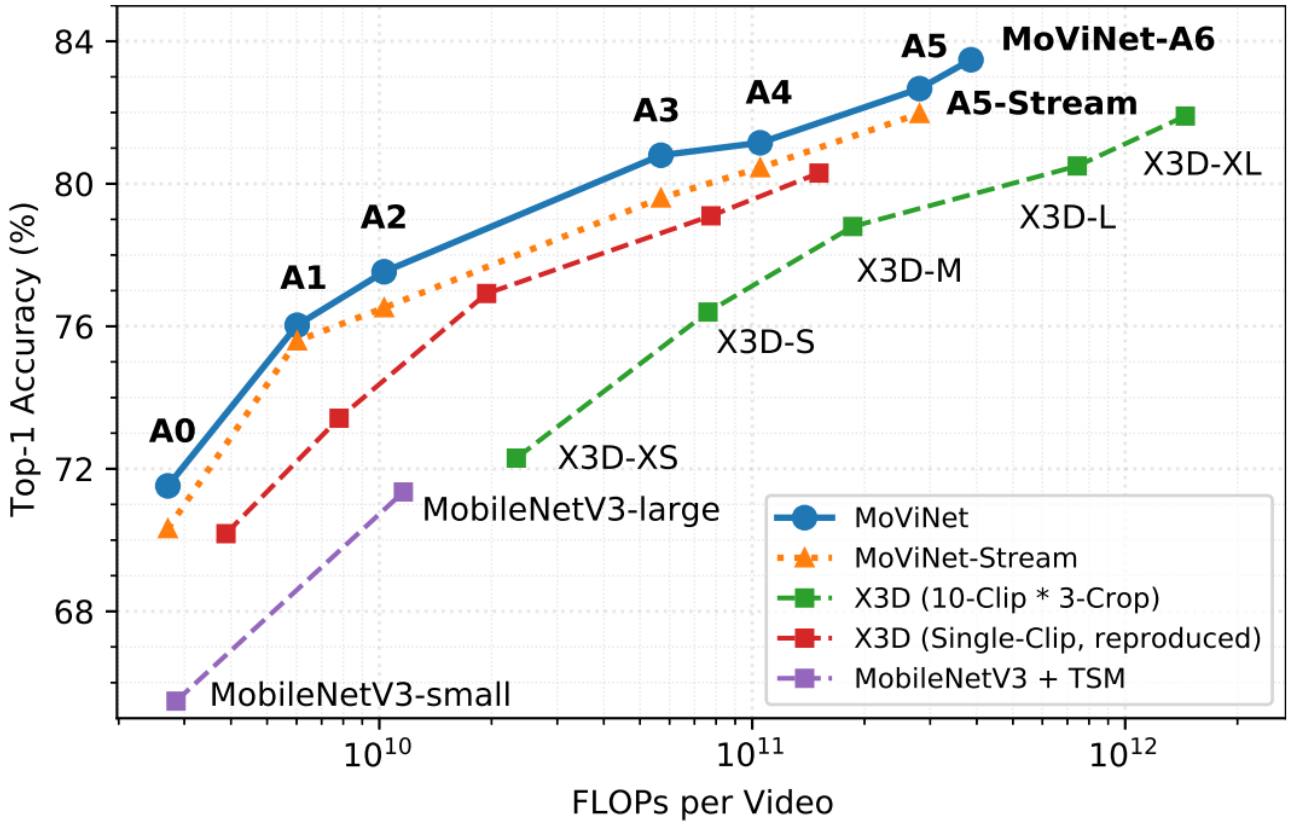


Figure modified and taken from [45] showing the accuracy vs. FLOPs tradeoff on Kinetics 600 [46]

MoViNets [45] – mobile & fast video recognition

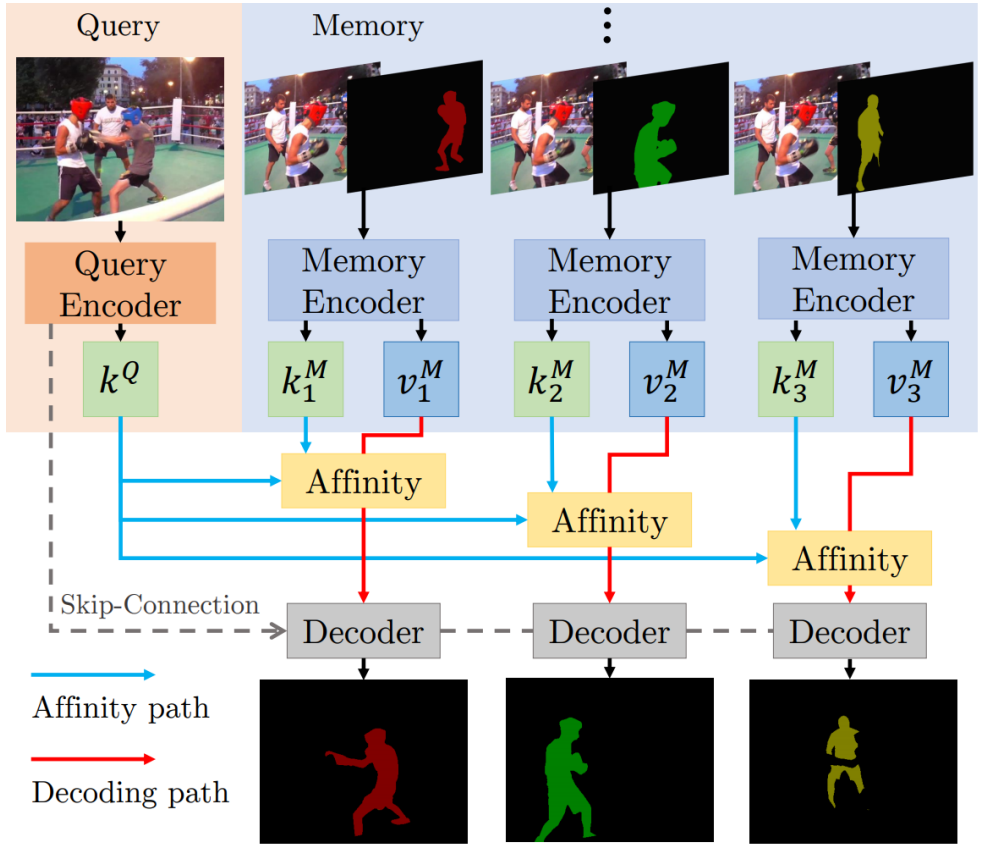
1. NAS: search space of model configurations (number of layers, kernel size, configuration of blocks etc.) searched with an objective function – MobileNetV3 [47] popularized it for ARM devices (i.e. mobile phone CPUs), MoViNets builds on MobileNetV3 search space expanded with 3D convs.

2. Multi-clip evaluation reduces memory to $O(T^{clip})$ but temporal receptive field only for a subclip and overlap computations are redundant – stream buffer B instead stores content of b frames: $B_{i+1} = (B_i \oplus \boldsymbol{x}_i^{clip})_{[-b:]} \rightarrow F_i = f\left(B_i \oplus \boldsymbol{x}_i^{clip}\right) \quad (14, 15)$

3. Stream buffer leads to minor accuracy drop – ensembling regains this: two MoViNets are trained at half frame rate, for inference one input is offset by one frame and arithmetic mean (before softmax) used

STCN [49] – space-timed object segmentation

- STM [48] (space-time memory) semi-supervised networks with target-specific memory banks
- **STCN** [49]: **correspondences are target-agnostic** and learn affinity with one RGB key frame



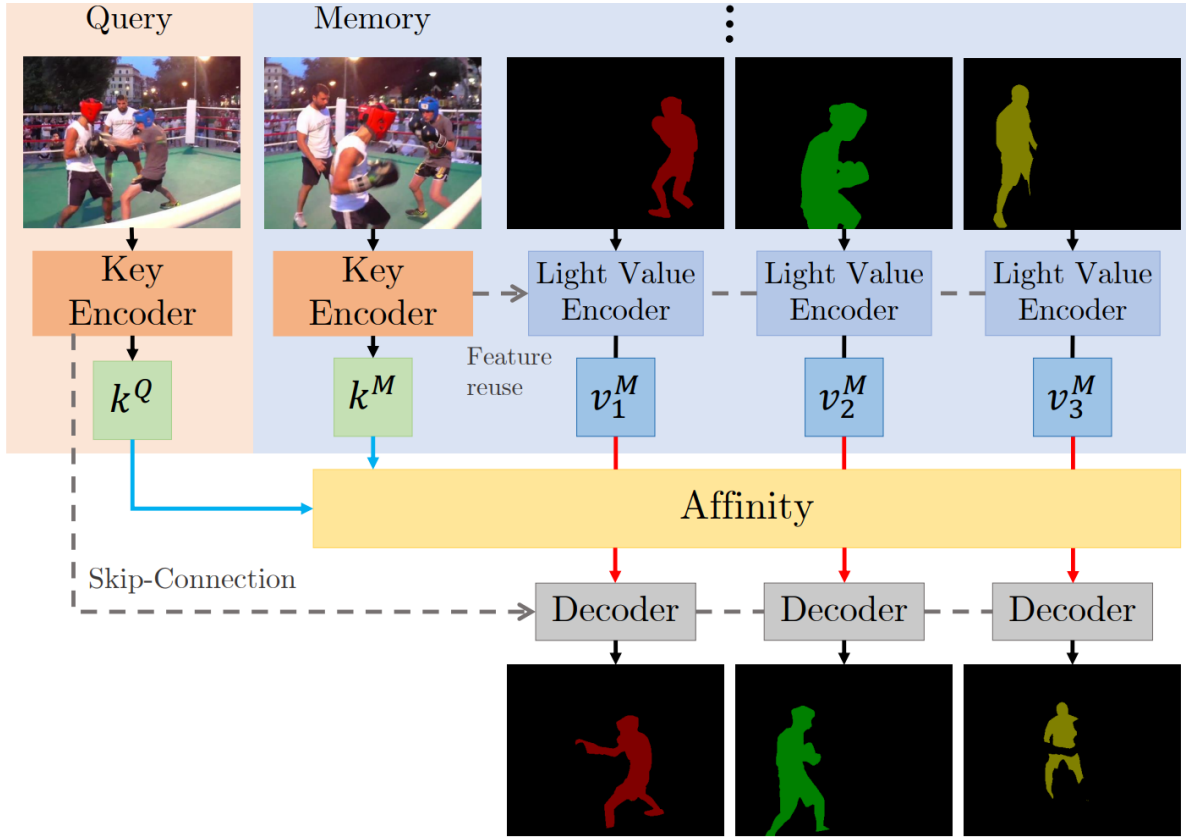


Figure taken from & showing the framework of [49]

STCN [49] – space-timed object segmentation

- Dot-product can be implemented very efficiently for affinity matrix computation

$$\boldsymbol{S}_{ij}^{dot} = \boldsymbol{k}_i^M \cdot \boldsymbol{k}_j^Q \Rightarrow \boldsymbol{S}^{dot} = (\boldsymbol{k}^M)^T \boldsymbol{k}^Q \quad (16)$$

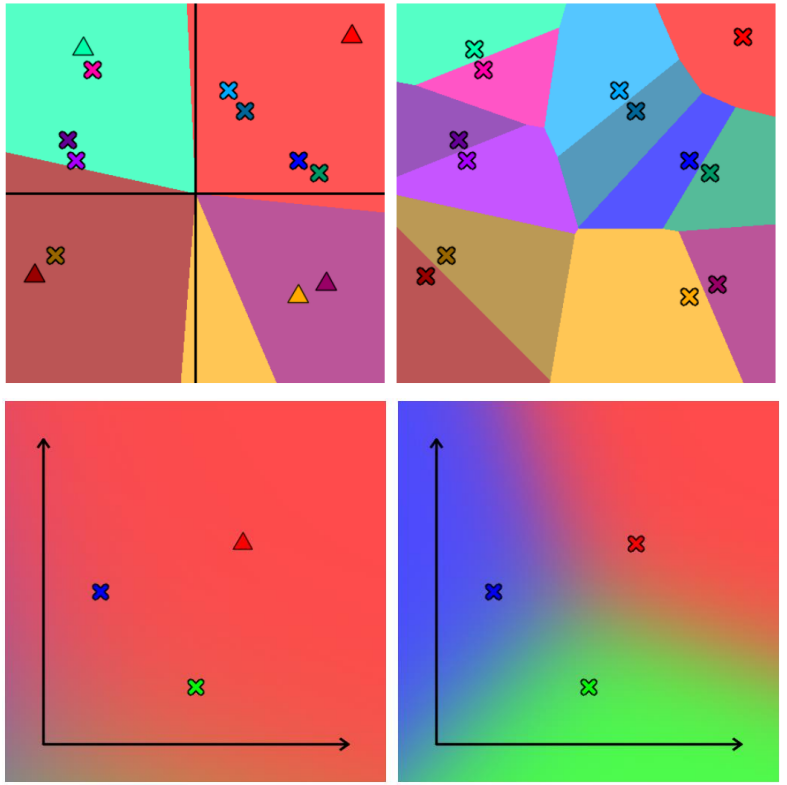
  but can lead to undesirable effects since high-magnitude „confident“ points dominate affinities [49] supressing other memory nodes

- L2 similarity (negative squared Euclidean)

$$\boldsymbol{S}_{ij}^{L2} = -\left\| \boldsymbol{k}_i^M - \boldsymbol{k}_j^Q \right\|_2^2 \quad (17)$$

  creates Voronoi diagrams and can be decomposed for easier implementation [64]

$$\boldsymbol{S}_{ij}^{L2} = 2\boldsymbol{k}_i^M \cdot \boldsymbol{k}_j^Q - \left\| \boldsymbol{k}_i^M \right\|_2^2 - \left\| \boldsymbol{k}_j^Q \right\|_2^2 \quad (18)$$



Upper figure showing regions colored as „most similar point“ under Dot-product (l) and L2 (r) – lower figure showing softmax similarities. Both figures taken from [49]

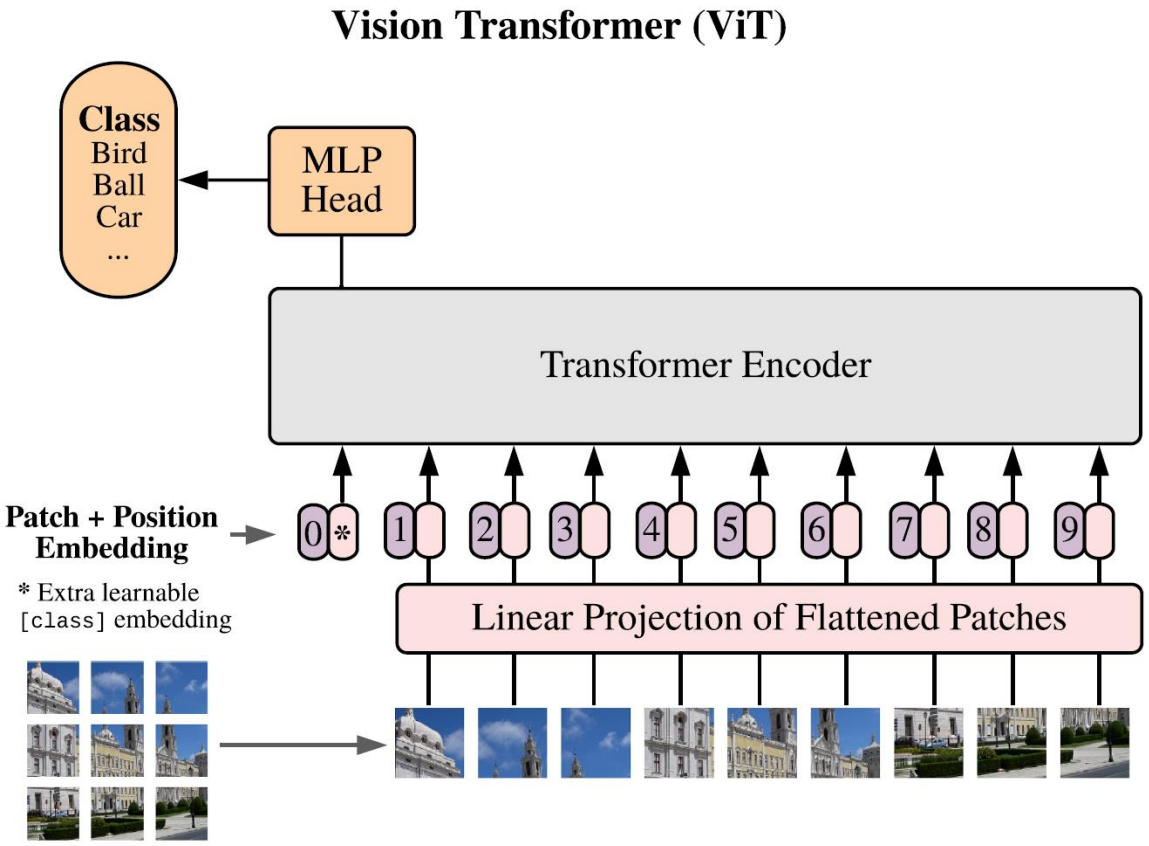
Transformers

- Convolutions in CNNs go back to the age-old bias-variance dilemma [50]:
  - Local connectivity: size of convolutional kernel determines receptive field (3x3 or 7x7) - lack of global understanding, i.e. dependencies between features
  - [52] argue that these limitations are beneficial in small-data regimes but in big-data settings where „all“ can be learned from samples its a drawback

- **Transformers (1900+ preprints since 2021/08: git-repo)**
  - Taken from natural language processing (NLP) – embeds words in codebook
  - Two main ideas [53]:
    - **Self-attention** to capture long-range relations between embeddings, factorize data distribution with attention – optimize with maximum likelihood
    - **Pre-training** on large (un)labelled corpus in (un)supervised manner

Transformers in computer vision

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale [54]:



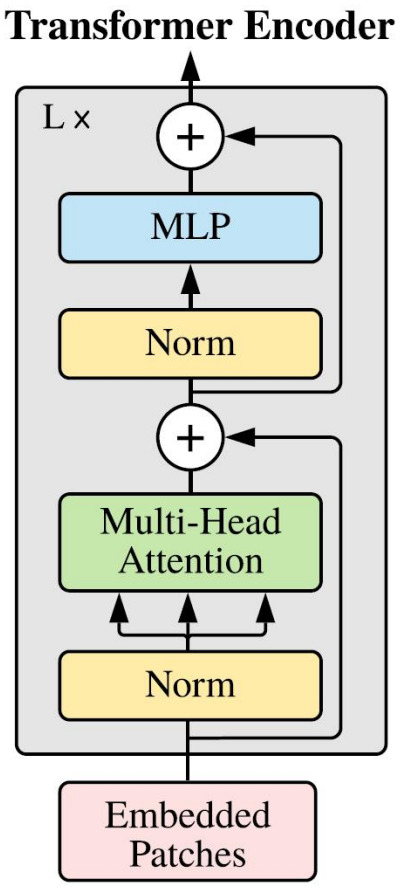


Image taken from [54] where the illustration of the transformer encoder was inspired by [55]

1. Image is split into patches
2. Embedded with position
3. Fed into transformer
4. Attention & dependencies
5. Classification with learnable “classification token”

TimeSformer [52] – attention for video classification

- Convolution free architecture based on [52] extended to space-time 3D volume
- Video clips are viewed as a sequence of 16x16 patches
- Analyze different attention schemes – divided space-time (middle column) performed best

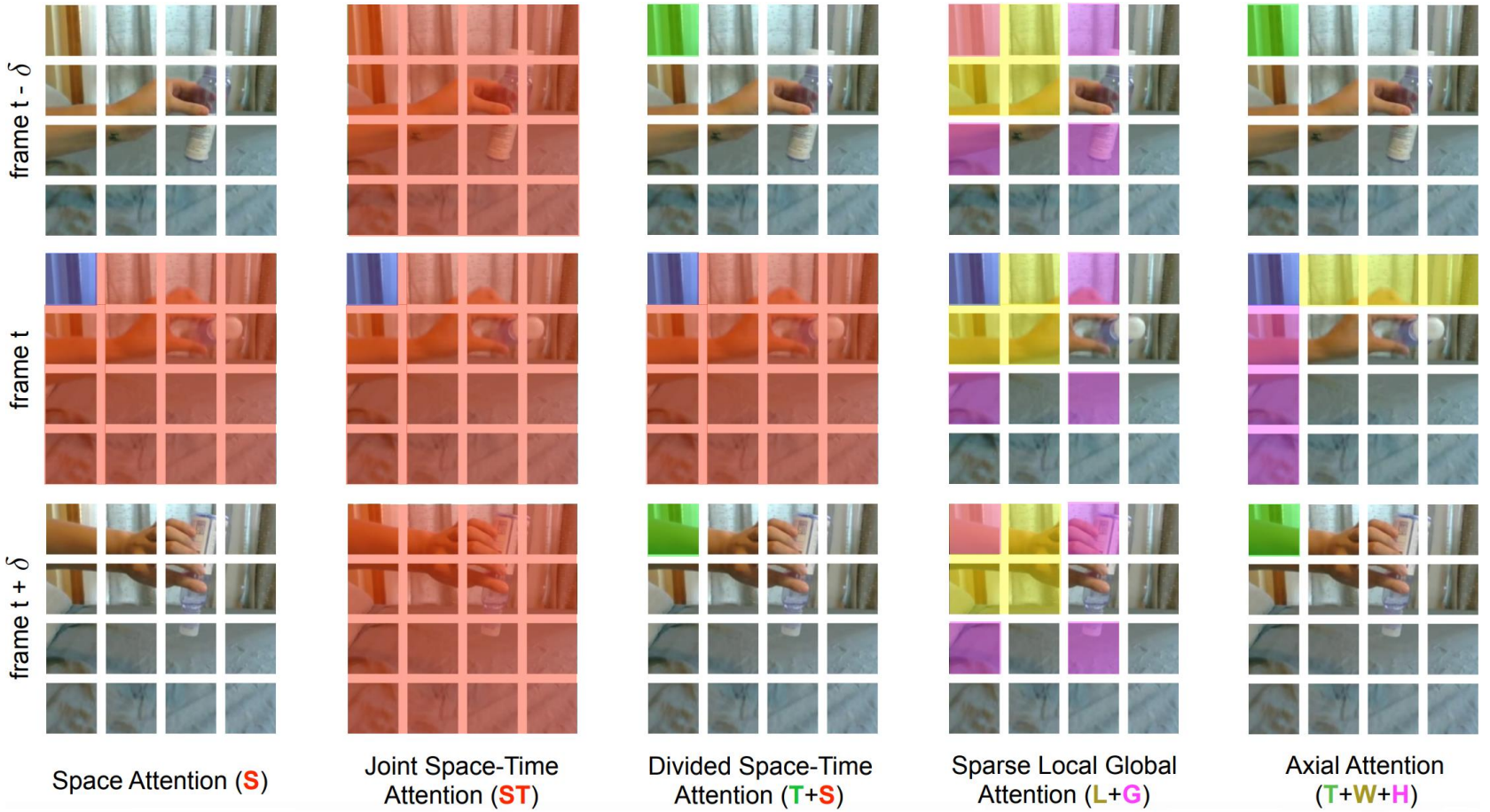
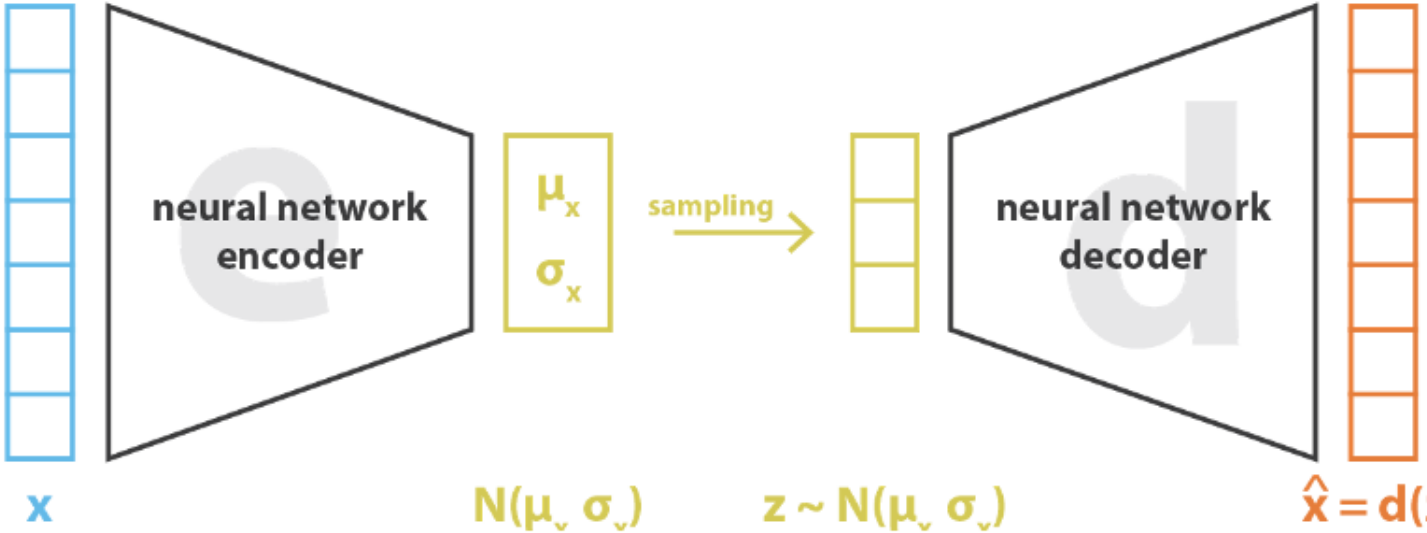
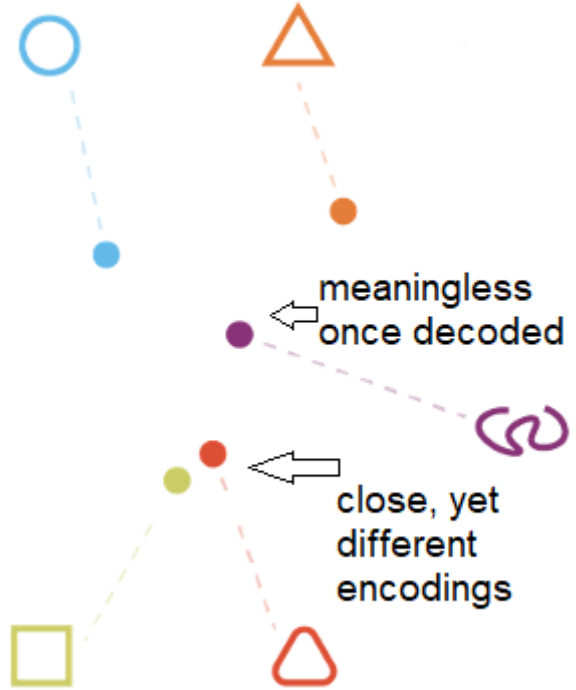


Figure taken from and showing the various attention schemes of [52]

VAEs – Variational Autoencoders

- The latent space in AE is irregular – we want continuity and completeness
- VAEs output mean and std.-dev. vector – draw samples – a probabilistic problem
- From a VI (variational inference) perspective we want to maximize the *evidence lower bound* - by minimizing KL (Kullback-Leibler) divergence: $KL(q(z|x)||\, p(z))$





Top and right figure modified taken from [56]: VAE architecture and irregular latent space in AEs

VQ-VAEs (Vector Quantization-Variational AEs)

- VQ-VAEs: derivative with discrete latent space – for every decoded feature vector the closest, discrete learnable vector from a **codebook** is used instead
- The idea comes from [57] who argue that language is inherently discrete, represented as a sequence of symbols and images can be concisely described by language [58] – VQ-VAEs further avoid posterior collapse

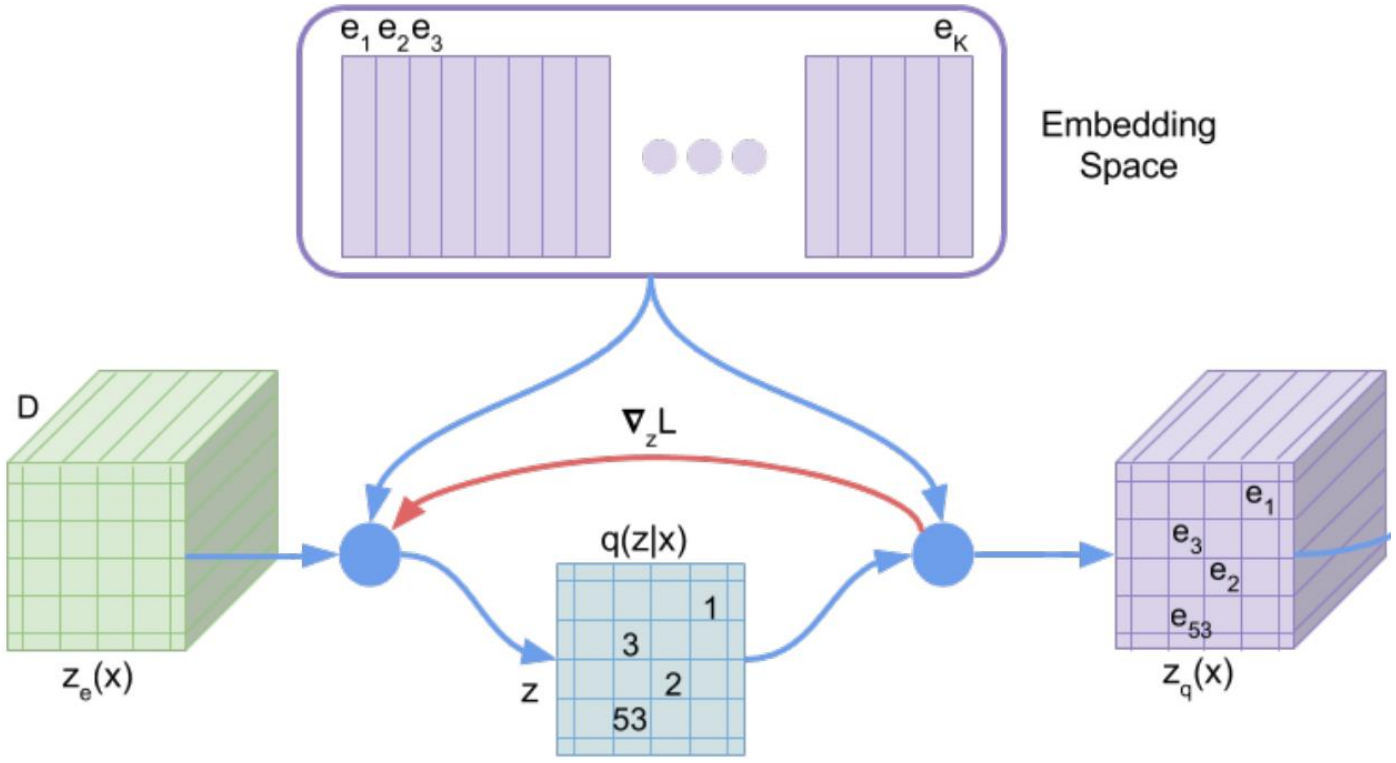


Figure modified and taken from [57] showing encoded feature vectors and codebook embedding

See [56] for a great explanation of VAEs and [59] for VQ-VAEs with PyTorch code

VideoGPT [60] – VQ-VAE & GPT for video generation

- VQ-VAE autoencoder uses 3D and transposed convolutions
- Encoder: learned downsampled discrete latents from raw pixels of video frames
- Transformer: Model autoregressive prior (dependent on past values) with latents
- Decoder: Sample from the prior and decode for video output

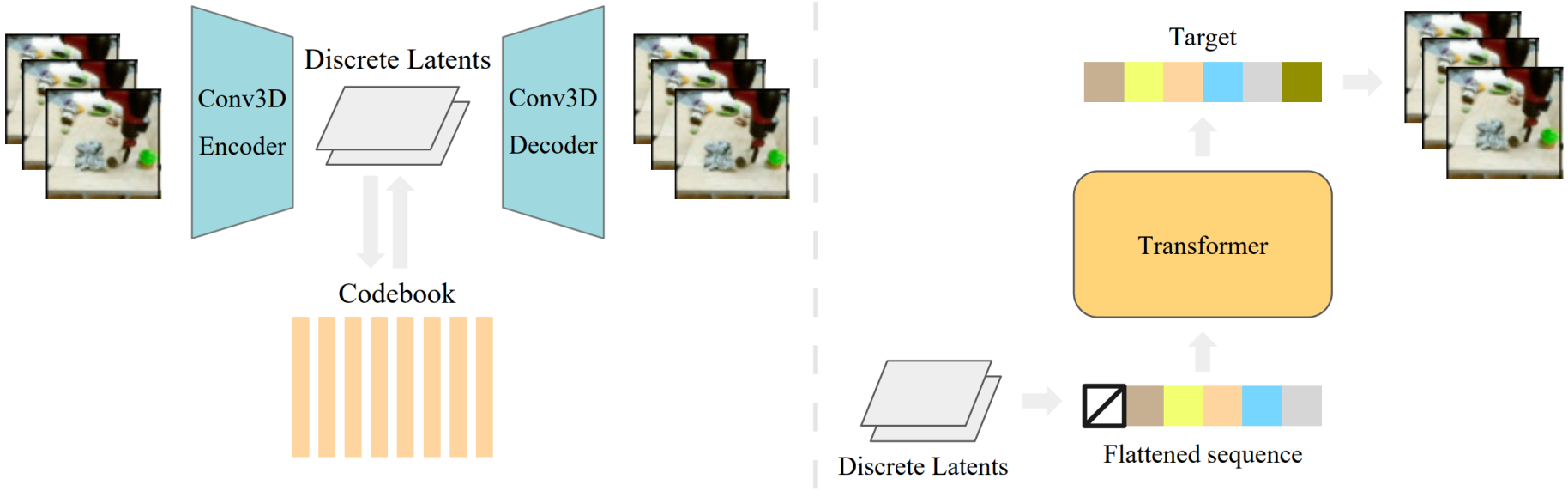


Figure taken from [60] showing the VQ-VAE on the left and the GPT (transformer) on the right

VideoGPT [60] – VQ-VAE & GPT for video generation

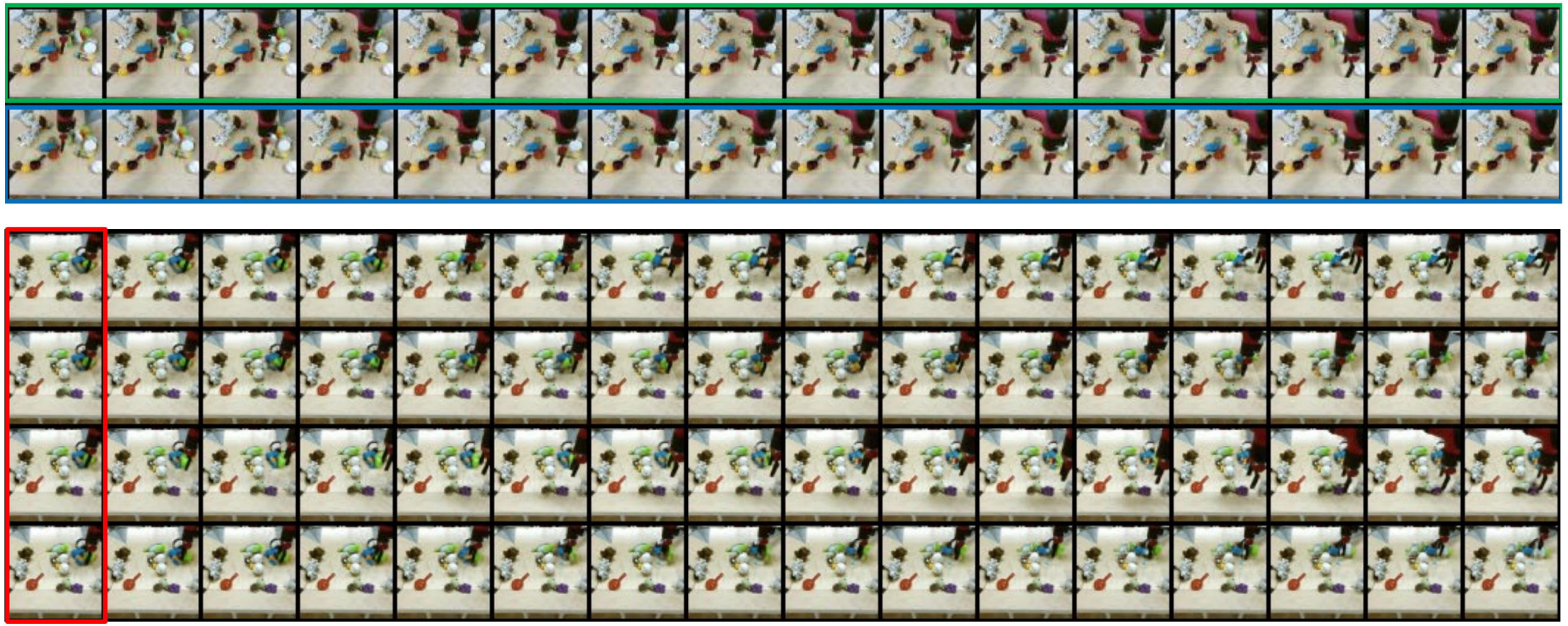
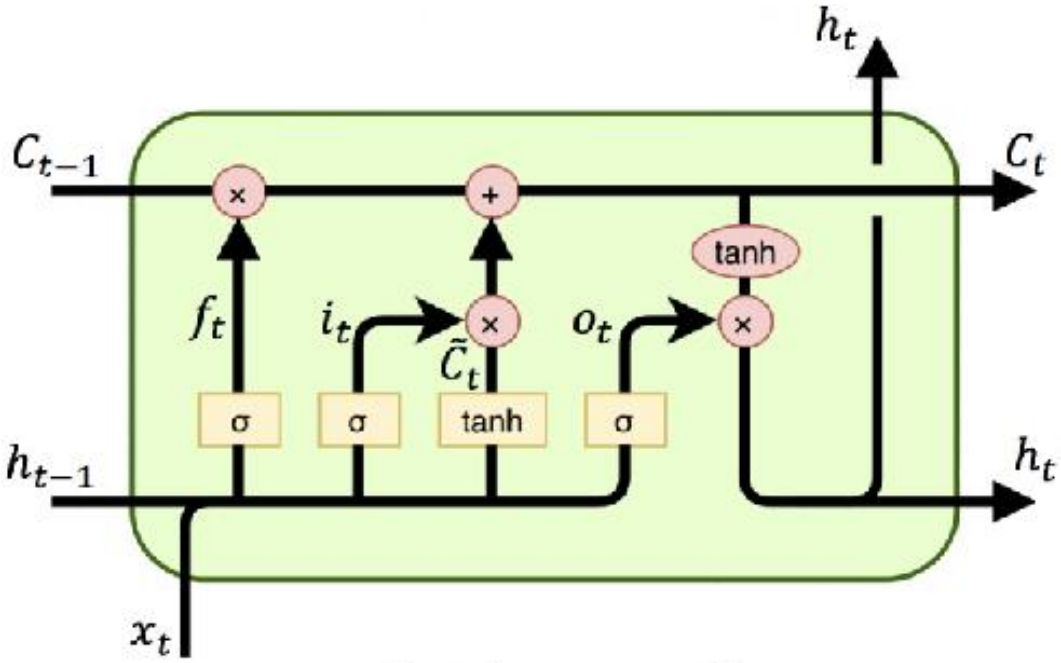


Figure taken from [60] showing VQ-VAE reconstructions on the top and generations on the bottom

LSTMs

- LSTMs proposed in [16] and [28]

- x_t... input at t
- h_{t-1}...hidden state at t-1
- C_{t-1}... Cell state at t-1
- $\tilde{C}$ ... Cell state candidate
- U ... weight matrix for the input
- W ... internal weight matrix for state
- i, f, o ... input, forget and output gate controlled by σ



$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$$
$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$$
$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$$
$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$$
$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$$
$$h_t = \tanh(C_t) * o_t$$

Figure modified and taken from [29]

PredRNNv2 [61] – RNN for video prediction

- LSTM [16] or ConvLSTM [62] do not model short-term spatial apperance changes and long-term multi-frame dynamics – propose spatiotemporal LSTM (ST-LSTM)

1. Propose zigzag memory flow across layers vertically and states horizontally
2. Decouple memory states for different aspects of spatiotemporal variations
3. Novel reverse scheduled sampling for non-Markovian (long-term) dependencies

- Trained end-to-end fully unsupervised with objective:

$$L = \sum_{t=2}^{T+K} \left\| \hat{X}_t - X_t \right\|_2^2 + L_{decouple}$$

where X is the input tensor of observations over T timesteps – K is the length for the future sequence that is to be predicted

PredRNNv2 [61] – ST-LSTM blocks

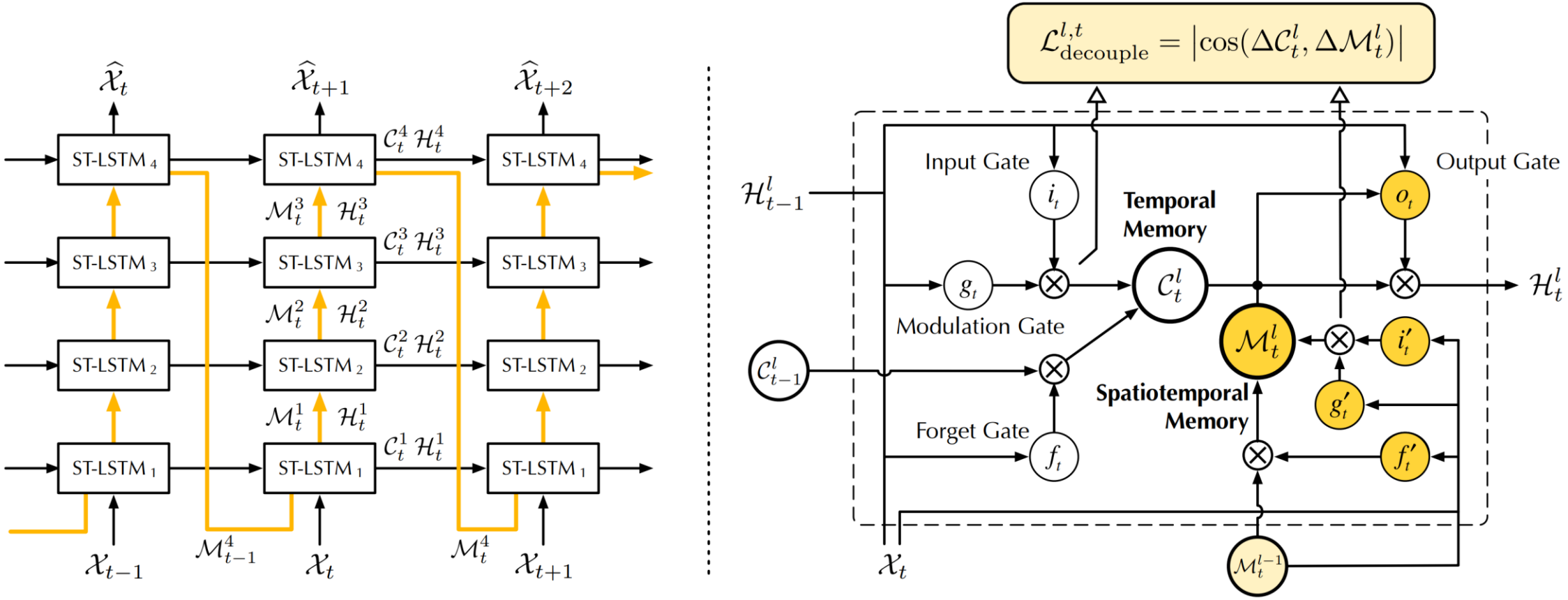
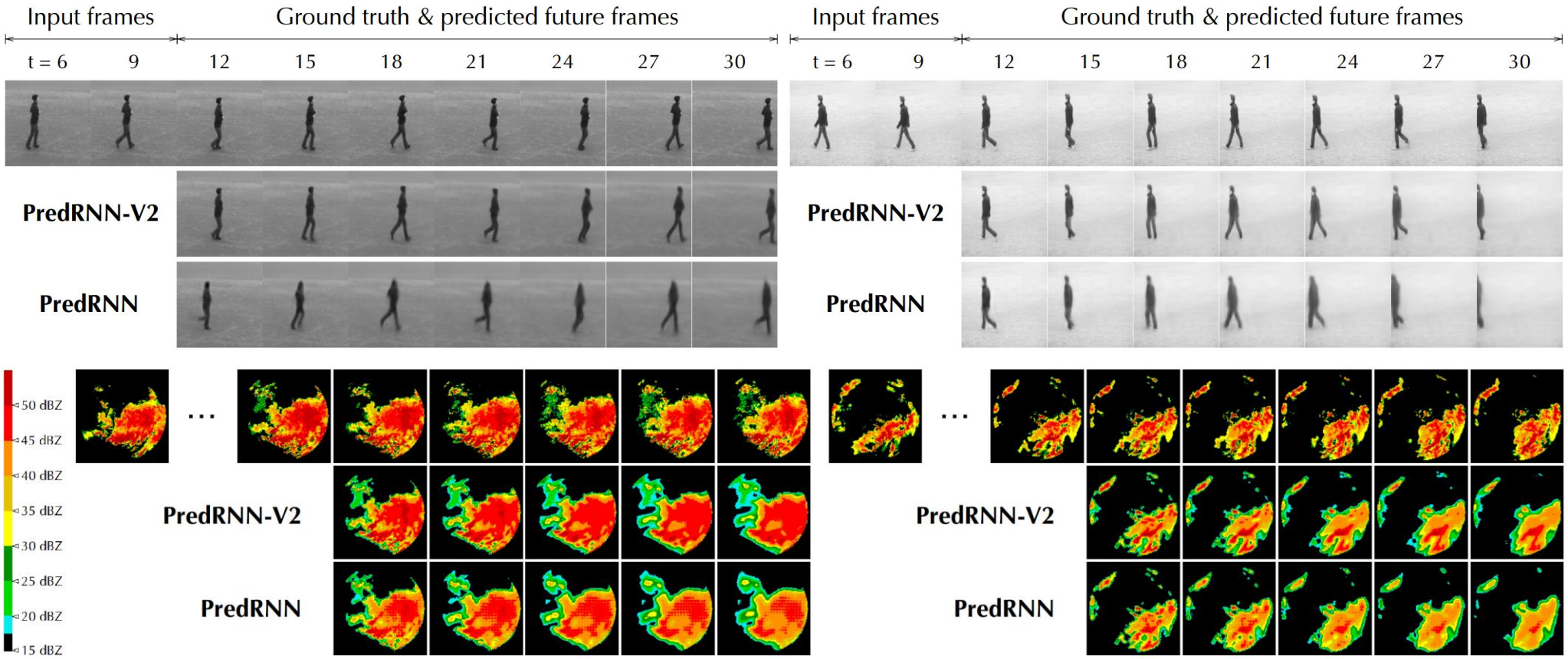


Figure taken from [61] showing a single ST-LSTM cell on the right (differences to ConvLSTM [62] in yellow circles) and the PredRNN main architecture on the left where orange arrows show the spatiotemporal memory flow

PredRNNv2 [61] – KTH and Radar echo results



Figures taken from [61] showing results on the KTH [63] and Radar Echo [61] test sets. See paper for detailed results.

Lessons learned

- Videos enable temporal perception beyond RGB images
- The redundancy is a blessing and curse – it enables many predictive tasks and self-supervision but also requires more efficient models
- 2D CNNs can be simply extended with a temporal dimension for classification using 3D CNNs
- For more complicated tasks, concepts like key-frames, memory banks, vision transformers, VAEs and RNNs have proven successful

Questions?

Suggestions?

Complaints?

Thank you for your attention

References

References marked with * have not been read by the lecturer (A. Kriegler) but are instead included for the sake of providing a historically accurate representation of the scientific progress in the field of deep learning. The correctness of this representation is largely based on works by J. Schmidhuber [2, 66].

[1] C.M. Bishop, *Pattern Recognition and Machine Learning,* Singapore: Springer, 2006. (PDF)

[2] J. Schmidhuber, "Deep learning in neural networks: An overview", *Neural networks,* 2015, 61, pp.85-117, 2015. (PDF)

[3] I. Goodfellow, Y. Bengio and A. Courville, *Deep learning*, Cambridge: MIT press, 2016. (e-Book)

[4] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach,* Fourth edition, New Jersey: Pearson, 2021. (accompanying website)

[5] M. Bronstein, J. Bruna, T. Cohen and P. Velickovic, "Geometric Deep Learning", GDL course part of AMMI 2021, 2021. [Online]. Available at: https://geometricdeeplearning.com/lectures/ (Accessed: 22.10.2021).

[6] T. Gärtner, M. Thiessen and A. Sepliarskaia, "194.100 Theoretical Foundations and Research Topics in Machine Learning". [Online]. Available at: https://preview.tinyurl.com/yf73xbu7 (Accessed: 11.05.2021).

[7] M. Charikar and C. Ré, "CS229: Machine Learning". [Online]. Available at: http://cs229.stanford.edu (Accessed: 11.05.2021).

[8] L. FeiFei, K. Ranjay, X. Danfei, „CS231n: Convolutional Neural Networks for Visual Recognition". [Online]. Available at: http://cs231n.stanford.edu (Accessed: 11.05.2021).

[9] L. Fridman, „MIT Deep Learning and Artificial Intelligence Lectures". [Online]. Available at: https://deeplearning.mit.edu (Accessed: 11.05.2021).

[10] G. Sanderson, „Deep Learning Series". [Online]. Available at: https://preview.tinyurl.com/m92b8r9u (1st part) (Accessed: 11.05.2021).

[11] U. Von Luxburg, „Mathematics for Machine Learning". [Online]. Available at: https://preview.tinyurl.com/jk9p3m9x (Accessed: 11.05.2021).

[12] L. Von Rueden, S. Mayer, R. Sifa, C. Bauckhage and J. Garcke, „Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions", *International Symposium on Intelligent Data Analysis (IDA)*, 2020, pp.548-560, 2020. (PDF)

[13] J. Duchi, „Introduction to Convex Optimization for Machine Learning", *Practical Machine Learning, Fall 2009*, UC Berkeley, 2009. (PDF)

[14] A. Ivakhnenko and V.G. Lapa, *Cybernetic predicting devices,* New York: CCM Information Corp., 1965. (accompanying website)

[15] J. Schmidhuber, "Learning Complex, Extended Sequences using the Principle of History Compression", *Neural Computation*, 1992, 4(2), pp.234-242, 1992. (PDF)

References

[16] S. Hochreiter, *Untersuchungen zu Dynamischen Neuronalen Netzen*. München: Technische Universität München (master‘s thesis, german), 1991. (PDF)

[17] A. Krizhevsky, I. Sutskever and G.E. Hinton, “Imagenet Classification with Deep Convolutional Neural Networks”, *Advances in Neural Information Processing Systems (NIPS)*, 2015, 25, pp.1097-1105, 2015. (PDF)

[18] W. Fedus, B. Zoph and N. Shazeer, “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity”. [Online]. Available at: https://arxiv.org/abs/2101.03961 (Accessed: 11.05.2021).

[19] Original source unknown. Taken from: L. Langer, „Algorithm Journey from PC to IoT – A Signal Processing Pipeline with TensorFlow 2.X on a Smartwatch“. [Online]. Available at: https://preview.tinyurl.com/5xfkchpb (Accessed: 11.05.2021).

[20] Ž. Ivezić, A.J. Connolly, J.T. Vanderplas and A. Gray, *Statistics, Data Mining and Machine Learning in Astronomy,* New Jersey: Princeton University Press, 2014. (accompanying website)

*[21] K. Fukushima and S. Miyake, “Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition”, *Biological Cybernetics,* 1980, 36, pp. 193-202, 1980. (PDF)

[22] S. Saha, “A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way”. [Online]. Available at: https://preview.tinyurl.com/6d8njrv6 (Accessed: 11.05.2021).

[23] Aphex34, „typical CNN architecture“. [Online]. Available at: https://preview.tinyurl.com/33xvmpfm (Accessed: 11.05.2021).

[24] M.D. Zeiler and R. Fergus, „Visualizing and Understanding Convolutional Networks“, *European Conference on Computer Vision*, 2014, pp.818-833, 2014. (arXiv preprint)

[25] C. Kyunghyun, B.V. Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, „Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation“, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp.1724-1734, 2014. (PDF)

[26] J. Schmidhuber, „Our A.I.‘s Impact on the World‘s 5 Most Valuable Public Companies“. 2017. [Online]. Available at: https://preview.tinyurl.com/ayfex2m9 (Accessed: 02.08.2022).

[27] C. Junyoung, Ç. Gülçehre, C. Kyunghyun, Y. Bengio, „Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling“, *NIPS 2014 Workshop on Deep Learning*, 2014. (PDF)

References

[28] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, *Neural Computation*, 1997, 9(8), pp.1735-1780, 1997. (PDF)

[29] S. Varsamopoulos, K. Bertels and C.G. Almudever, “Designing neural network based decoders for surface codes”, *Neural Networks for Fast Surface Code Decoding*, 2018. (PDF)

[30] C. Olah, “Understanding LSTM Networks”, [Online]. Available at: https://preview.tinyurl.com/ftapu35z (Accessed: 02.08.2021).

[31] H. Bandyopadhyay, “An Introduction to Autoencoders: Everything You Need to Know”, [Online]. Available at: https://preview.tinyurl.com/474f7n6k (Accessed: 09.08.2021).

[32] V. Campos, “Video Analysis”, UPC Barcelona 2018, [Online]. Available at: https://preview.tinyurl.com/dzceafpc (Accessed: 09.08.2021).

[33] I. Misra, C.L. Zitnick and M. Hebert, “Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification”, *Europen Conference on Computer Vision (ECCV)*, pp.527-544, 2016. (PDF)

[34] D. Pathak, R.B. Girshick, P. Dollar, T. Darrell, B. Hariharan, “Learning Features by Watching Objects Move”, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.6024-6033, 2017. (PDF)

[35] R. Arandjelovic, A. Zisserman, “Look, Listen and Learn”, *2017 IEEE International Conference on Computer Vision (ICCV)*, pp.609-617, 2017 (PDF)

[36] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga and G. Toderici, “Beyond Short Snippets: Deep Networks for Video Classification”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4694-4702, 2015. (PDF)

[37] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko and T. Darrell, “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), pp. 677-691, 2017. (PDF)

[38] D. Tran, L.D. Bourdev, R. Fergus, L. Torresani and M. Paluri, “Learning Spatiotemporal Features with 3D Convolutional Networks”, *2015 IEEE International Conference on Computer Vision (ICCV)*, pp.4489-4497, 2015. (PDF)

[39] C. Feichtenhofer, A. Pinz and R. Wildes, “Spatiotemporal Residual Networks for Video Action Recognition”, *Advances In Neural Information Processing Systems (NIPS)*, pp.3468-3476, 2016. (PDF)

[40] J. Carreira and A. Zisserman, “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset”, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.4724-4733, 2017. (PDF)

[41] K. Simonyan and A. Zisserman, “Two-Stream Convolutional Networks for Action Recognition in Videos”, *Neural Information Processing Systems (NIPS)*, pp.568-576, 2014. (PDF)

References

[42] E. Shelhamer, K. Rakelly, J. Hoffman and T. Darrell, “Clockwork Convnets for Video Semantic Segmentation”, 2016. (arXiv-report)

[43] J. Carreira, V. Patraucean, L. Mazare, A. Zisserman and S. Osindero, “Massively Parallel Video Networks”, *European Conference on Computer Vision (ECCV)*, pp.649-666, 2018. (PDF)

[44] V. Campos, B. Jou, X. Giro-i-Nieto, J. Torres and S.-F. Chang, “Skip RNN: Learning to Skip State Updates in Recurrent Neural Networks”, *International Conference on Learning Representations (ICLR)*, 2018. (PDF)

[45] D. Kondratyuk, L. Yuan, Y. Li, L. Zhang, M. Tan, M. Brown and B. Gong, “MoViNets: Mobile Video Networks for Efficient Video Recognition”, *2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.16020-16030, 2021. (PDF)

[46] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier and A. Zisserman, “A Short Note about Kinetics-600“, *ArXiv*, 2018. (PDF)

[47] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le and H. Adam, “Searching for MobileNetV3”, *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp.1314-1324, 2019. (PDF)

[48] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim. “Video object segmentation using space-time memory networks”. *International Conference on Computer Vision (ICCV)*, pp.9226-9235. (PDF)

[49] H. K. Cheng, Y.-W. Tai and C.-K. Tang, “Rethinking Space-Time Networks with Improved Memory Coverage for Efficient Video Object Segmentation”, *ArXiv*, 2021. (PDF)

[50] S. Geman, E. Bienenstock and Rene Doursat, „Neural Networks and the Bias/Variance Dilemma“, *Neural Computation*, 4(1), 1-58, 1992. (PDF)

[51] J. Cordonnier, A. Loukas and M. Jaggi, „On the relationship between self-attention and convolutional layers“, *8th International Conference on Learning Representations*, ICLR 2020, 2020. (PDF)

[52] G. Bertasius, H. Wang and L. Torresani, „Is Space-Time Attention All You Need for Video Understanding?“, *ArXiv*, 2021. (PDF)

[53] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan and M. Shah, „Transformers in Vision: A Survey“. [Online]. Available at: https://arxiv.org/abs/2101.01169 (Accessed: 11.05.2021).

[54] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly and J. Uszkoreit, „An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale“. [Online]. Available at: https://arxiv.org/abs/2010.11929 (Accessed: 11.05.2021).

References

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uskoreit, L. Jones, A.N. Gomez, L. Kaiser and I. Polosukhin, „Attention Is All you Need“, *Proceeding of the 31st International Conference on Neural Information Processing (NIPS)*, 2017, pp.6000-6010, 2017. (PDF)

[56] J. Rocca, „Understanding Variational Autoencoders (VAEs)“. [Online]. Available at: https://preview.tinyurl.com/5rfvhb78 (Accessed: 16.08.2021).

[57] A. van den Oord, O. Vinyals and K. Kavukcuoglu, “Neural Discrete Representation Learning”, *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, pp.6309-6318, 2017. (PDF)

[58] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, “Show and tell: A neural image caption generator”, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.3156-3164, 2015. (PDF)

[59] A. Gordić, „VQ-VAEs: Neural Discrete Representation Learning | Paper + PyTorch Code Explained“. [Online]. Available at: https://preview.tinyurl.com/mc2auz8u (Accessed: 16.08. 2021).

[60] W. Yan, Y. Zhang, P. Abbeel and A. Srinivas, „VideoGPT: Video Generation using VQ-VAE and Transformers“. [Online]. Available at: https://arxiv.org/abs/2104.10157 (Accessed: 30.08.2021).

[61] Y. Wang, H. Wu, J. Zhang, Z. Gao, J. Wang, P.S. Yu and M. Long, „PredRNN: A Recurrent Neural Network for Spatiotemporal Predictive Learning“. [Online]. Available at: https://arxiv.org/abs/2103.09504 (Accessed: 30.08.2021).

[62] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong and W.-C. Woo, „Convolutional LSTM Network: a machine learning approach for precipitation nowcasting“, *NIPS 15: Proceedings of the 28 International Conference on Neural Information Processing Systems*, pp.802-810, 2015. (PDF)

[63] C. Schüldt, I. Laptev and B. Caputo, „Recognizing human actions: a local SVM approach“, *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, 3, pp.32-36, 2004. (PDF)

[64] H. Kim, G. Papamakarios and A. Mnih, „The Lipschitz Constant of Self-Attention“. [Online]. Available at: https://arxiv.org/pdf/2006.04710.pdf (Accessed: 30.08.2021).

[65] Saama, „Different Kinds of Convolutional Filters“. [Online]. Available at: https://preview.tinyurl.com/2p98v4ks (Accessed: 26.11.2021) (Original image source unknown)

[66] J. Schmidhuber, „Critique of 2018 Turing Award for Drs. Bengio & Hinton & LeCun. 2020“. [Online]. Available at: https://preview.tinyurl.com/3vmfemxh (Accessed: 11.05.2021).