Multimodal Computational Methods in Political Science

OCR, CNN fine-tuning and Vision Transformers

Clara Fochler

Ludwig-Maximilians-Universität München

June 16, 2026

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Agenda

Updates on Last Session

Optical Character Recognition

CNN fine-tuning

Transformers

Vision Transformers (ViT)

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Updates on Last Session: Hashing & Data Augmentation

RandomCrop / RandomFlip

- These transforms **do not create duplicates** — they crop/flip images *in place* within the dataset
- You can set a **probability** for how often an image is flipped or cropped

What is a Hash?

- A hash converts an image into a **unique digital fingerprint**
- Before hashing, we pre-process the image so that **similar (but not identical)** images can still be detected

Further Reading

- Stack Overflow: differences between hashing methods
  `https://stackoverflow.com/questions/998662/what-is-image-hashing-used-for`
- Official package website: `https://www.phash.org/`

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Optical Character Recognition (OCR)

OCR is the automated conversion of images containing text into machine-readable text.
We will not dive into the mathematics here - this is a completely own research field with many different methods.

- **Pre-processing is essential:** Clean images (thresholding, morphological ops) dramatically improve OCR accuracy
- **OCR engines are not image processors:** –> they need clean, segmented input to perform well
- **Post-processing matters:** Apply spellcheck, regex, and domain heuristics to correct inevitable OCR errors
- **Key tools:** Tesseract (most popular), pytesseract (Python wrapper), and cloud-based OCR APIs

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

CNN Fine-Tuning

CNNs are typically pretrained on large datasets (e.g., ImageNet with 1.3M images and 1000 classes) and then adapted to new tasks.

- **Pretraining:** models like AlexNet, VGG, ResNet, DenseNet
- **Transfer learning:** reuse learned visual features
- **Fine-tuning:** replace final layer for new classes (e.g., binary output)
- **Optional:** freeze early layers, retrain upper layers only
- **Benefit:** less data + faster training + strong performance

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

CNN Hyperparameters

Hyperparameters are settings chosen *before training* that control how the CNN learns - they control how fast, stable and generalizable learning is

Core training hyperparameters:

- **Epochs:** number of full passes over training data
- **Learning rate:** step size for weight updates (too high $\rightarrow$ unstable, too low $\rightarrow$ slow learning)
- **Batch size:** number of samples per update (speed vs. stability trade-off)
- **Train/validation split:** prevents overfitting via held-out evaluation

Regularization and optimization:

- **Dropout:** randomly deactivates neurons to reduce overfitting
- **Optimizer (e.g., SGD):** defines how gradients update weights
- **Momentum:** smooths updates using past gradients

Learning rate scheduling:

- **Step size:** how often learning rate is reduced
- **Gamma:** factor controlling decay strength

Diagnostics

Does the CNN generalize well beyond the training data?

Standard Evaluation Metrics

(already covered in Part 1 of this seminar)

- **Accuracy:** overall correctness
- **Precision:** correctness of positive predictions
- **Recall:** ability to find all true positives
- **F1 score:** balance of precision and recall

Overfitting Detection

- Training loss keeps decreasing
- Validation loss stagnates or increases $\rightarrow$ model memorizes instead of generalizing

Model Debugging Strategies

- Inspect false positives / false negatives
- Use confusion matrix for error patterns
- Add diverse training data if bias is detected

Search Strategy

Use **Bayesian**, **Random**, or **Grid Search** to efficiently explore the hyperparameter space and avoid overfitting.

Transformers (encoder-only)

Self-Attention & Positional Encoding

- **Self-Attention:** each token generates queries, keys, and values to attend to *all* tokens simultaneously $\rightarrow$ enables parallel computation, but incurs $O(n^2)$ complexity (prohibitive for long sequences)
- Also called *attention pooling*
- **Positional Encoding:** self-attention has no inherent notion of token order $\rightarrow$ absolute or relative position information must be explicitly injected

Basic Structure of the Encoder

- Several **identical layers** arranged sequentially
- Each layer: **multi-head self-attention** + **position-wise feed-forward network**
- **Residual connections** around both sublayers, followed by **layer normalization**
- Output: each token $\rightarrow$ a vector representation

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How Attention Weights are Computed

Compatibility (similarity score)

$$a(q, k_i)$$

Measures how relevant key k_i is for query q.

Normalization via Softmax

α_i = softmax(a(q, k_i)) $= \frac{\exp(a(q,k_i))}{\sum_{j=1}^{m} \exp(a(q,k_j))}$

α_i converts raw similarity scores into probabilities + ensures $\sum_i \alpha_i = 1$ + highlights the most relevant elements

Note: $a(q, k_i)$ is a placeholder here - Transformers use a specific scaled dot-product version of it ($q^\top k_i / \sqrt{d}$), since the dot product is a simple and efficient way to measure vector similarity. The softmax + weighted sum framework above stays the same. - if you are interested why you can have a look at the source I assigned for today and read chapter "11.3. Attention Scoring Functions"

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Attention as Weighted Aggregation

Attention computes a weighted combination of value vectors based on similarity between a query and keys.

Attention Formula:

$$\text{Attention}(q, \mathcal{D}) = \sum_{i=1}^{m} \alpha_i v_i$$

- $\mathcal{D} = \{(k_1, v_1), \ldots, (k_m, v_m)\}$: database of key-value pairs
- q: query (what we are looking for)
- k_i: keys (descriptions of items)
- v_i: values (information content)
- α_i: how relevant each item is to the query

$\rightarrow$ mixture of value vectors, weighted by relevance to the query

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Efficient Attention

Attention must work efficiently with variable-length sequences and large minibatches $\rightarrow$ we need extra functions

Masked Softmax (handling padding)

- Real sequences have different lengths (e.g., NLP sentences)
- Shorter sequences are padded with `<blank>` tokens
- These should NOT affect attention
- Solution: mask invalid positions before softmax ($-\infty$)

$\rightarrow$ Attention weights for padding tokens become 0 + only valid tokens contribute to the output

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Attention in Seq2Seq Models (Bahdanau et al.)

The issue with Seq2Seq without attention:

- Encoder compresses the entire input sequence into a single fixed vector
- Decoder relies only on this fixed representation
- Leads to information bottleneck for long sequences

Attention idea:

- Instead of one fixed context, compute a **context per decoding step**
- Use attention over all encoder hidden states
- Dynamically select relevant parts of the input at each output step

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Multi-Head Attention (Extending Single Attention)

Instead of computing one attention, we compute **multiple attentions in parallel**.

- Linearly project queries, keys, and values into different subspaces
- Apply attention independently in each subspace (each is a **head**)
- Concatenate outputs from all heads
- Apply a final linear transformation

$\rightarrow$ Different heads can focus on different patterns + richer representations than single attention

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

The Transformer Architecture (Vaswani et al., 2017)

A sequence-to-sequence model built entirely on attention → no RNNs or CNNs.

Key Building Blocks

- **Self-attention:** dependencies between all tokens
- **Multi-head attention:** different representation subspaces
- **FFN:** per-token non-linear transformation
- **Residual + LayerNorm:** stabilizes deep training

Decoder-Specific

- **Encoder–decoder attention:** aligns output with input

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Transformers vs CNNs/RNNs

Key advantages vs. CNNs and RNNs:

- Fully parallel computation (no recurrence)
- Shortest path length between tokens $\rightarrow$ better long-range dependency learning
- But: computational cost is quadratic in sequence length

Transformers for Vision (ViT)

How can we use Transformers for images?
→ *treat image patches as tokens*

Image → Sequence of Patch Embeddings

- **Patching:** Split image into fixed-size patches (e.g., 16×16), flattened and linearly projected
- **Spatial Encoding:** Add learnable positional embeddings to retain spatial information
- **Classification token:** Prepend a learnable `[CLS]` token to aggregate global features

Architecture

- **Patch embedding:** Linear projection of image patches + addition of learnable positional embeddings
- **Transformer encoder:** Processing the patch sequence and the `[CLS]` token via multi-head self-attention to capture global dependencies
- **MLP head:** Using the aggregated representation of the `[CLS]` token for final classification

Limitations

- Self-attention is quadratic in patches → expensive for high-resolution images
- Needs large datasets to outperform CNNs

Discussion

Do you notice any remaining issues we did not have for CNNs?

Vision Transformers (ViT)

Positional Encoding:

- Self-attention has no built-in notion of order
- Positional encodings inject sequence order into token embeddings

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Thanks for your attention!

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

REFERENCES

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N. (2021). An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale. International Conference on Learning Representations (ICLR). https://arxiv.org/abs/2010.11929

Rosebrock, A. (2021, August 9). What is Optical Character Recognition (OCR)? PyImageSearch. https://pyimagesearch.com/2021/08/09/what-is-optical-character-recognition-ocr/

Webb Williams N, Casas A, Wilkerson JD. Images as Data for Social Science Research: An Introduction to Convolutional Neural Nets for Image Classification. Cambridge University Press; 2020.

Zhang, A., Lipton, Z. C., Li, M., Smola, A. J. (2024). Dive into deep learning. Cambridge University Press. https://d2l.ai