Transformers / Graph neural networks

Lecture 22 of “Mathematics and AI”

Outline

1. Attention (continued)
2. Transformers

   Neural word embeddings, multihead attention
3. Foundation models

   BERT, GPT
4. Outlook

   Graph neural networks, unsupervised learning

Attention (continued)

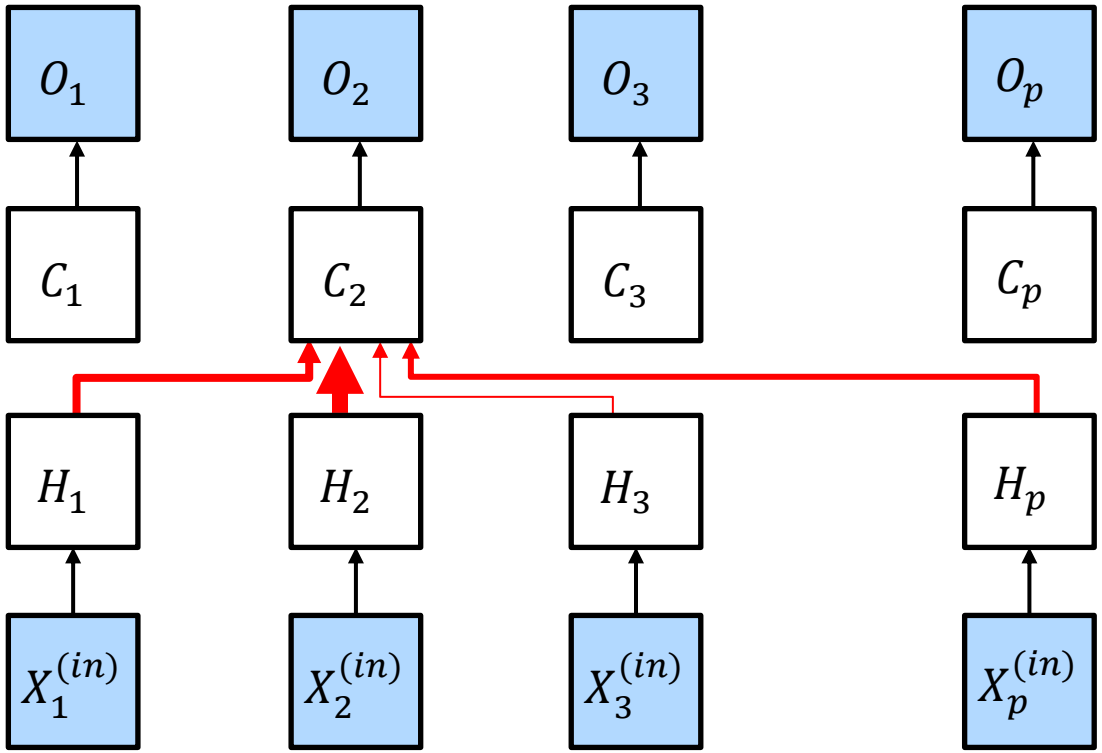
Self-attention

Output

Features with context added via self-attention mechanism

Embedded features

Input features



Weights are proportional to similarity measure between

Examples

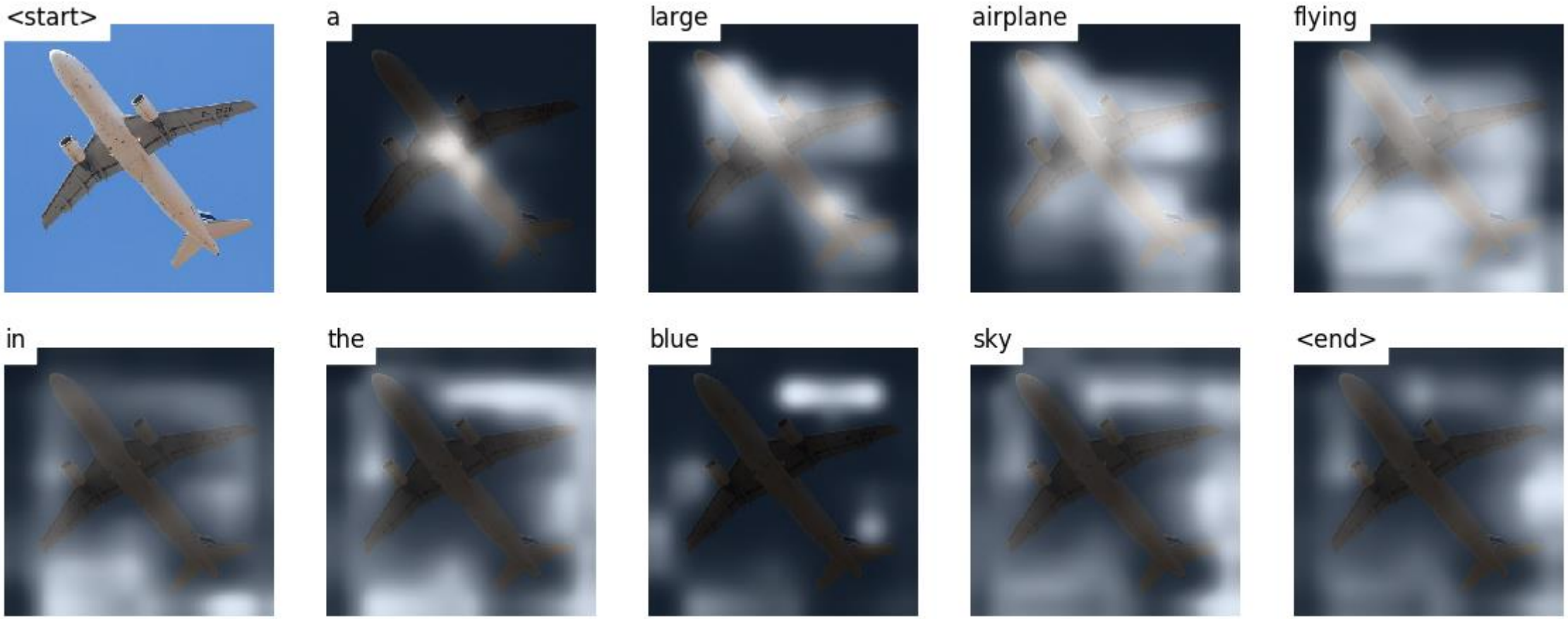


Image source: https://github.com/yunjey/show-attend-and-tell

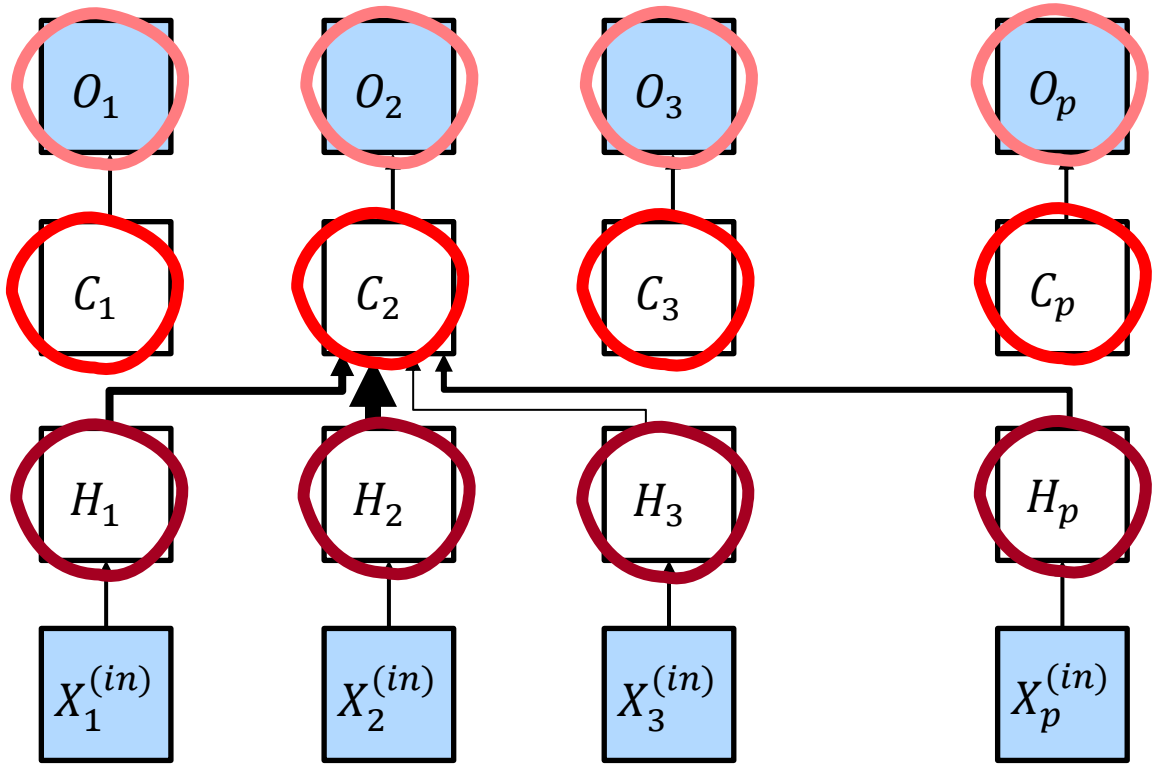
Forward pass for self-attention

Output

Features with context added via self-attention mechanism

Embedded features

Input features



Step 3

Step 2

Weights are proportional to similarity measure between

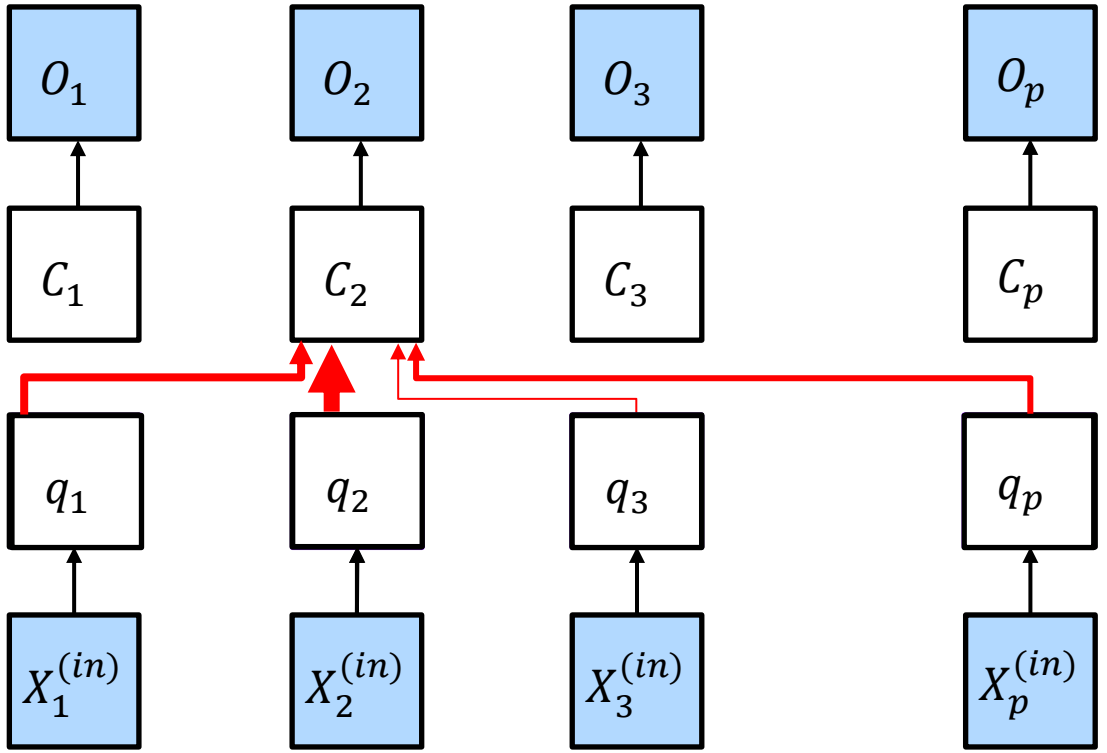
Step 1

Forward pass is parallelizable!!

Self-attention with QKV formulation

- Create 3 linear projections of $x_j^{(in)}$ using weights $W^{(key)}$, $W^{(value)}$, $W^{(query)}$
- Use queries, keys, and values
  - Queries: $q_j = W^{(query)} x_j^{(in)}$
  - Keys: $k_j = W^{(key)} x_j^{(in)}$
  - Values: $v_j = W^{(value)} x_j^{(in)}$
- Context for prediction O_2 is

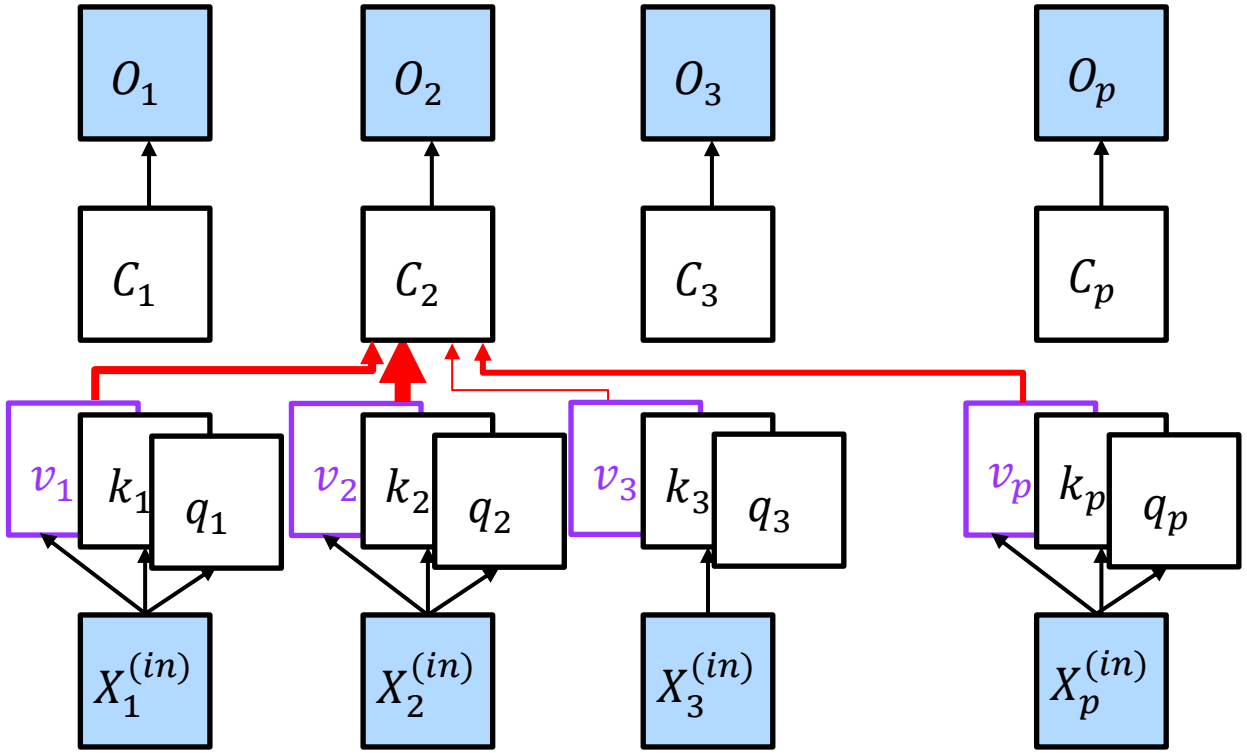
$$H_2^{(2)} = f\left(x_2^{(in)}, \sum_j softmax(a_{2,j}) v_j\right) \text{ with } a_{i,j} = q_i \cdot k_j$$



Self-attention with QKV formulation

- Create 3 linear projections of $x_j^{(in)}$ using weights $W^{(key)}, W^{(value)}, W^{(query)}$
- Use queries, keys, and values
  - Queries: $q_j = W^{(query)} x_j^{(in)}$
  - Keys: $k_j = W^{(key)} x_j^{(in)}$
  - Values: $v_j = W^{(value)} x_j^{(in)}$
- Context for prediction O_2 is

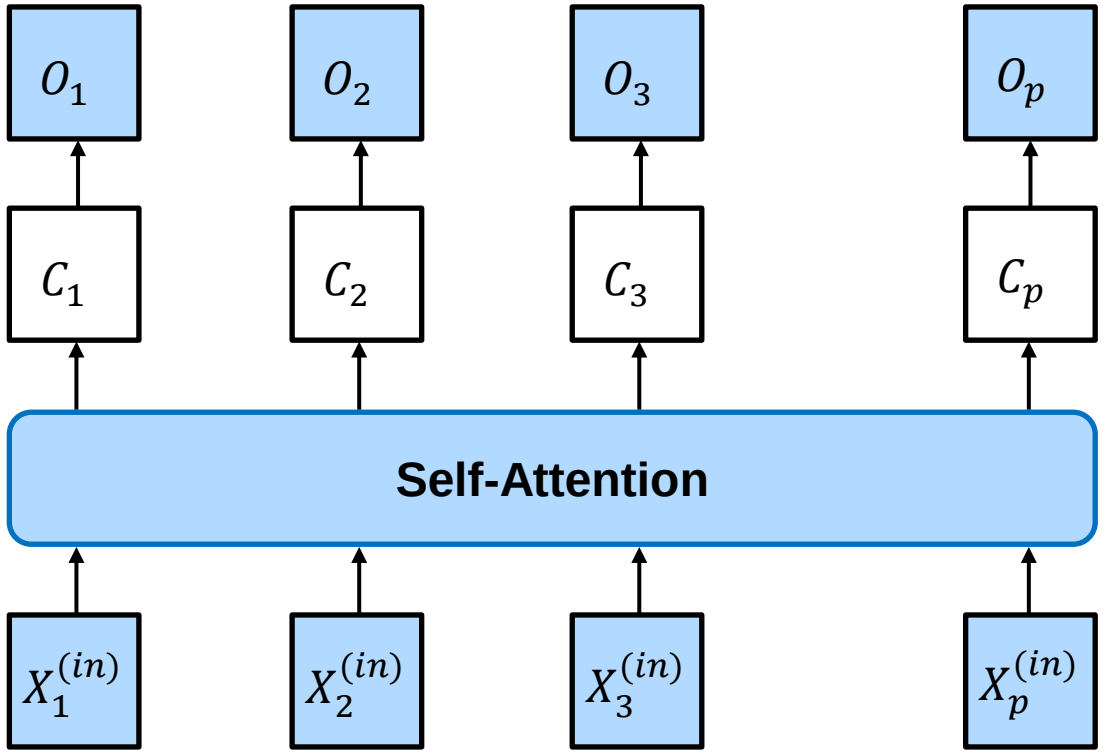
$$H_2^{(2)} = f\left(x_2^{(in)}, \sum_j softmax(a_{2,j}) v_j\right) \text{ with } a_{i,j} = q_i \cdot k_j$$



Self-attention with QKV formulation

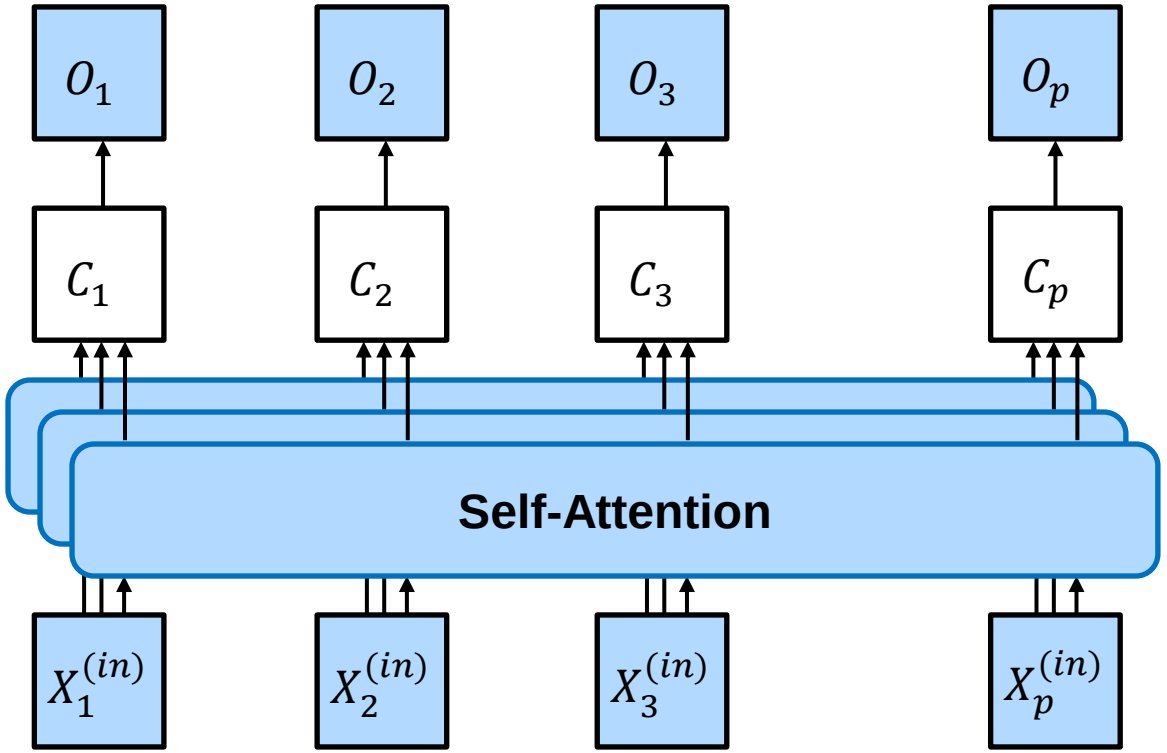
- Create 3 linear projections of $x_j^{(in)}$ using weights $W^{(key)}, W^{(value)}, W^{(query)}$
- Use queries, keys, and values
  - Queries: $q_j = W^{(query)} x_j^{(in)}$
  - Keys: $k_j = W^{(key)} x_j^{(in)}$
  - Values: $v_j = W^{(value)} x_j^{(in)}$
- Context for prediction O_2 is

$$H_2^{(2)} = f\left(x_2^{(in)}, \sum_j softmax(a_{2,j}) v_j\right) \text{ with } a_{i,j} = q_i \cdot k_j$$



Multiheaded self-attention (MHA)

- Several self-attention layers side-by-side
- Can learn several “sub-contexts” that are relevant to the context C_i
- Tends to lead to gain in training speed and training-data efficiency



Transformers

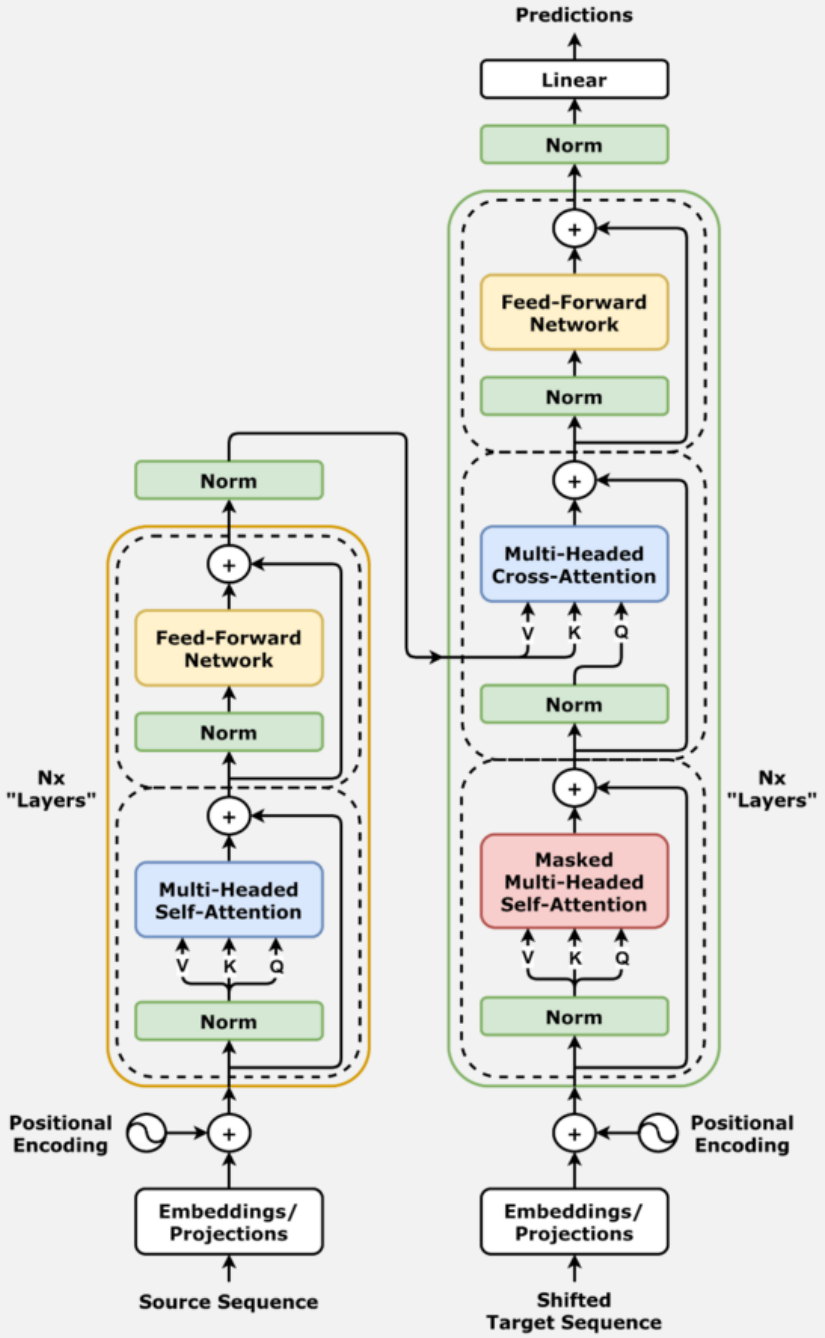


Image source: Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez*† University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Forward pass in a transformer

- Parallelizable training using self-supervised learning approach (“teacher forcing”)
- Sequential output generation

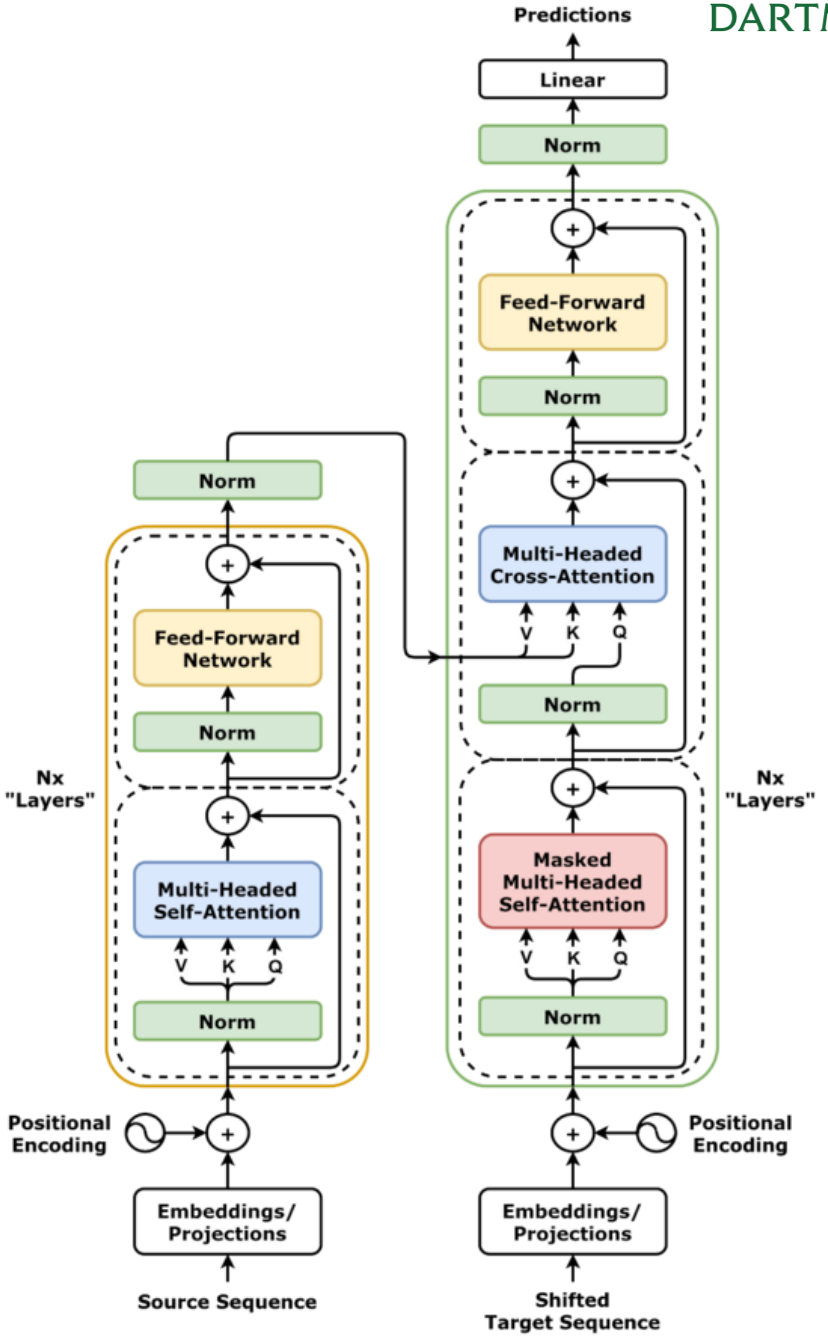


Image source: Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Foundation models and GPTs

Transfer learning

- Use knowledge (i.e., feature extraction/embedding) trained in one setting (task and/or data set) in another setting
- Transfer knowledge in the form of a fixed feature-extraction pipeline or as the initial state for the feature-extraction pipeline

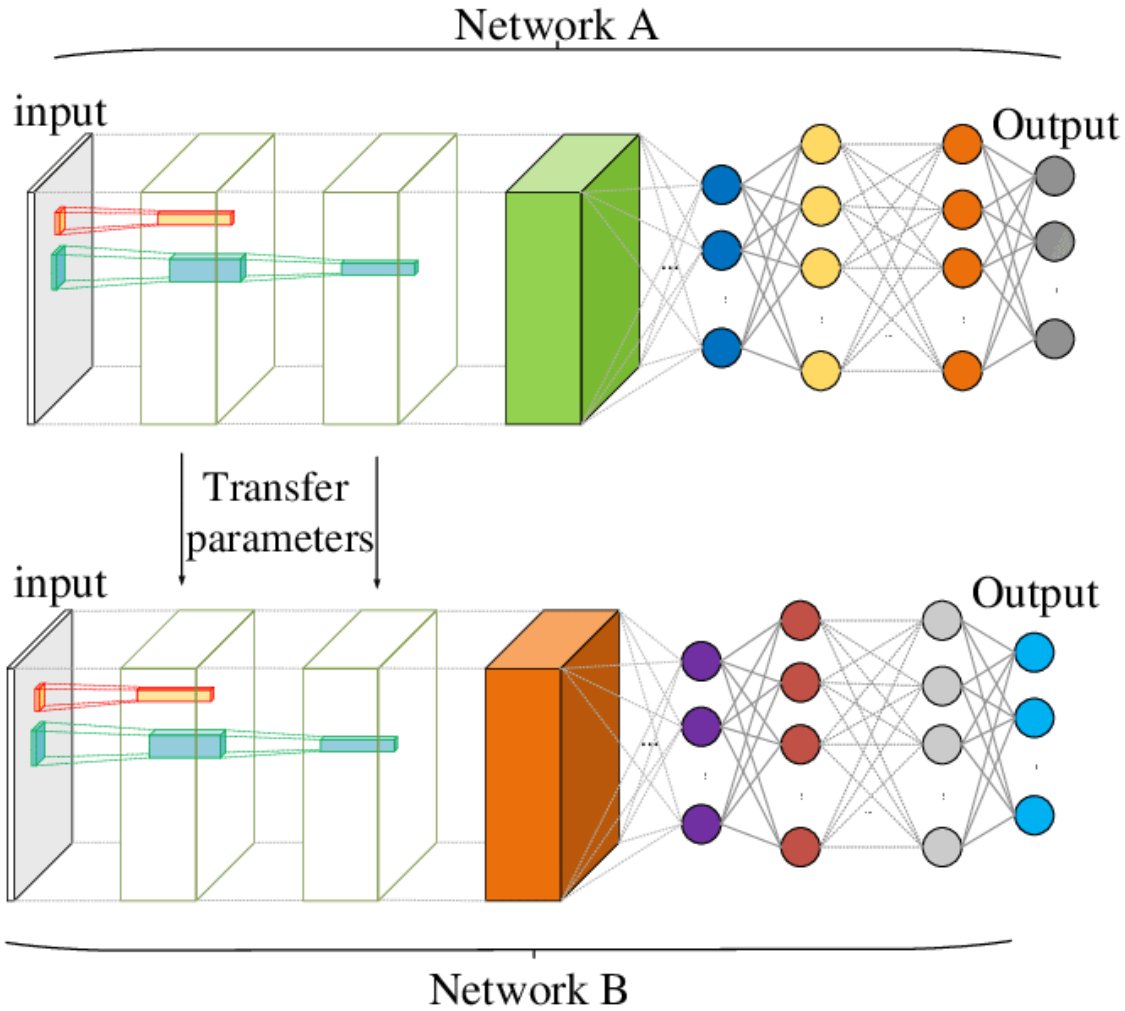


Image source: Lemley, Joseph & Bazrafkan, Shabab & Corcoran, Peter. (2017). Transfer Learning of Temporal Information for Driver Action Classification.

Foundation models

- Obtain a very large data set
- Train a very large network to (hopefully) yield a very good feature embedding
- Reuse this encoder for various tasks related to similar input/ output data
- Examples: Semantic embeddings for text or images

Image source: https://spiritandtruthonline.org/building-on-the-firm-foundation/

GPTs

- Generative Pretrained Transformer
- Use decoder of a pre-trained transformer to generate text
- Uses several word embeddings as foundation models
- Is a foundation model for various LLMOps

Image source: Reuters

Teaching transformers math

https://youtu.be/k9xLg-3WfG8?si=hCE7lycdMicdfnPm

Francois Charton: “Mathematics as a Translation Task - the Importance of Training Distributions”

- Machines tend to develop heuristics that work well for most observations
- Need to expose machines to the right math facts in the right frequency and order to steer them to learning the correct mathematical rules as opposed to an incorrect heuristic
- Distributions of training observations matter a lot!

Outlook

Challenges and successes in neural ML

Challenges

- Need large data sets
- Have large data sets
- Long training time
- Structured data

Successful innovations

- Architecture restrictions that mirror ***regular*** structures in input (CNN, RNN)
- Adapt to on ***semantic similarity*** in input (attention)
- Architecture restrictions that mirror ***irregular*** structures (GNN)
- Combinations of architectural restrictions and attention (e.g. GAT)

Graph neural networks

- Each node defines an input feature
- Embedding depends on network structur
- Graph-convolutional networks (GCN):
  - Neighboring nodes influence context
- Graph/attention transformer (GAT):
  - Similar neighboring nodes influence context most

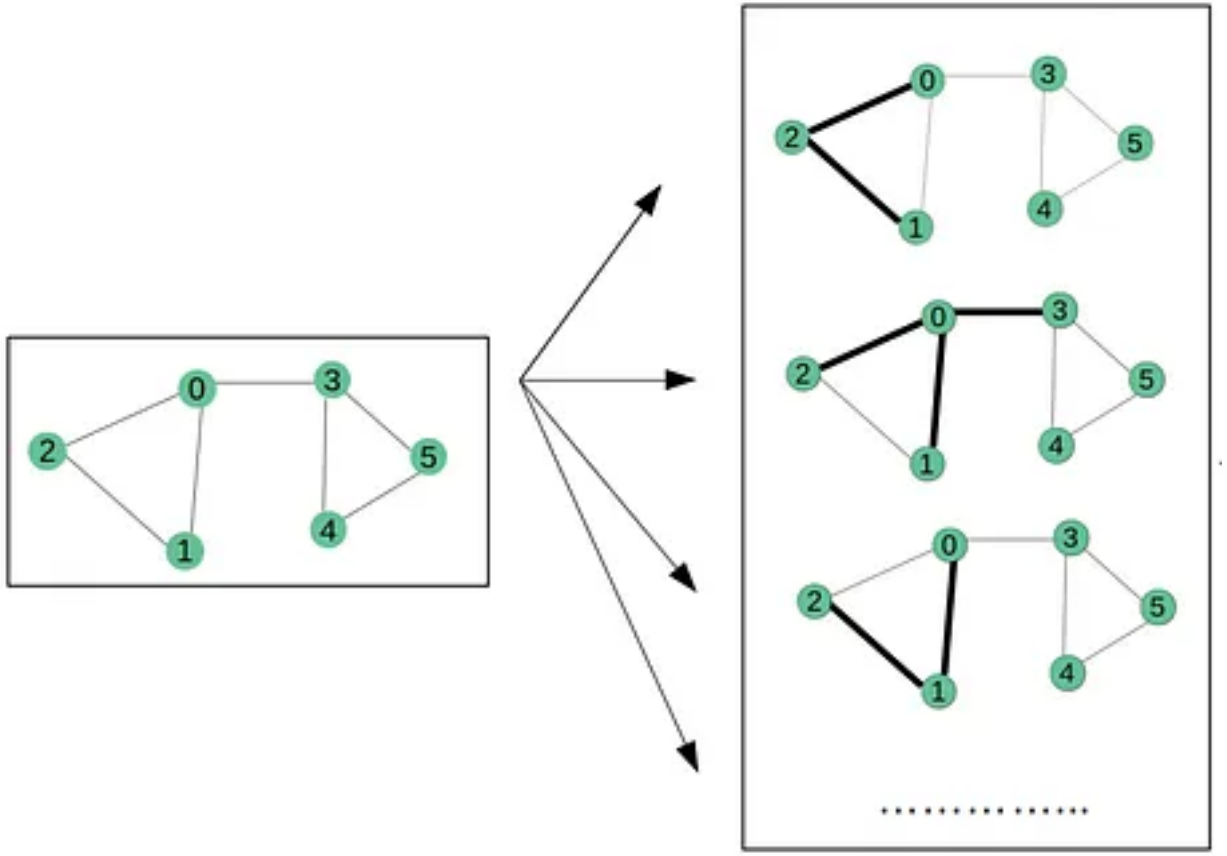
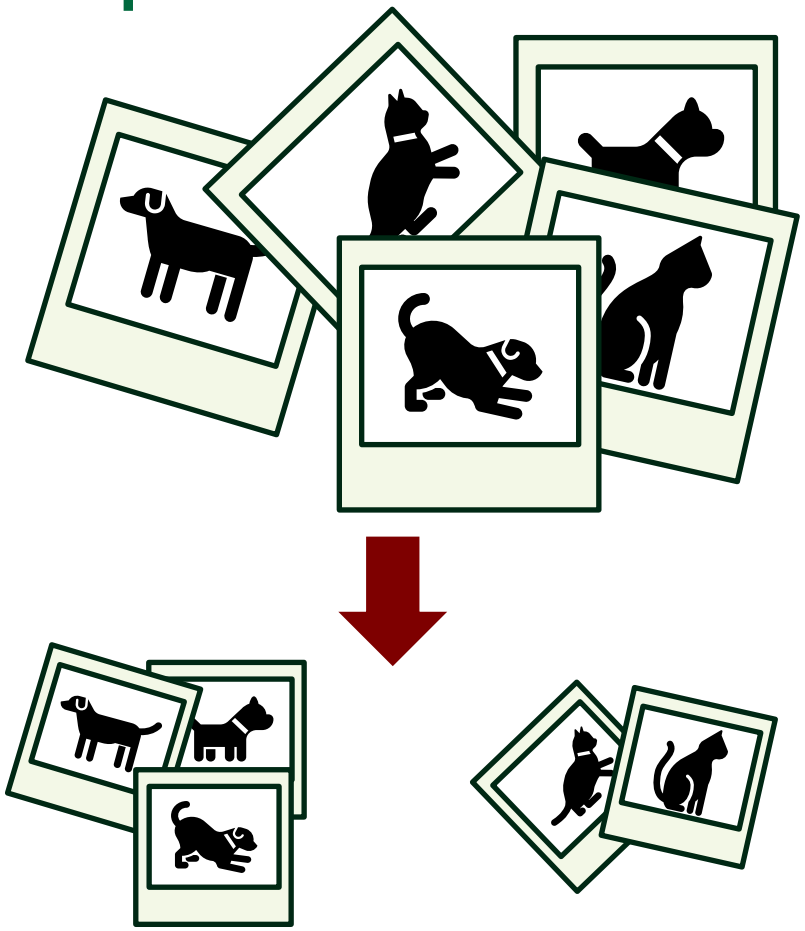


Image source: https://towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b

Another learning paradigm: Unsupervised learning

- Unsupervised learning: Learning ***unknown*** information from structured data
- Hard because no ground truth available
  - No output labels to compare to
  - No straightforward model validation
  - No straightforward hyperparameter fitting



Applications of unsupervised learning

- Data exploration
  - Extract salient descriptors of large data sets
- Data visualization
  - Visualize the most salient descriptors visually
- Feature engineering for supervised learning
  - Improve accuracy, training speed, and data efficiency of supervised learners
  - e.g., principal component regression (PCR)

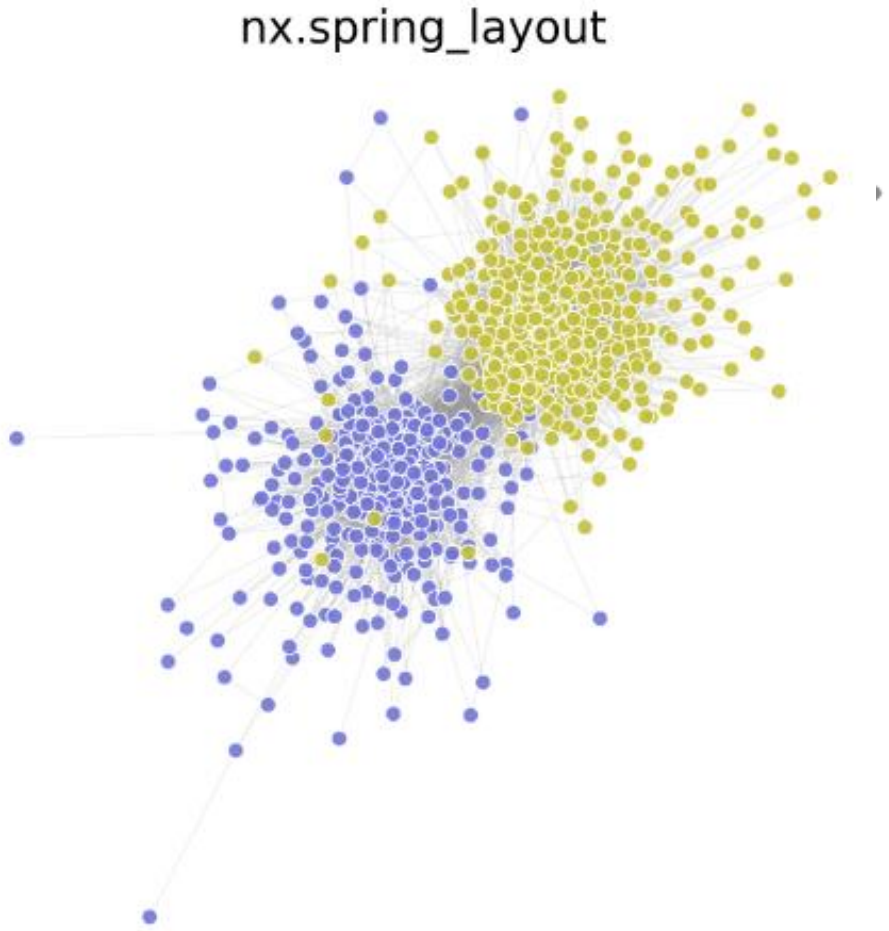


Image generated with code sourced from https://github.com/jkbren/networks-and-dataviz

Clustering via k-means

- Sort a data set into k subsets (i.e., “clusters)
- The data points in each cluster should be as close as possible to each other (and as far as possible from other clusters)
- Optimization problem

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg\min_{\mathbf{S}} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

- Solution is NP hard

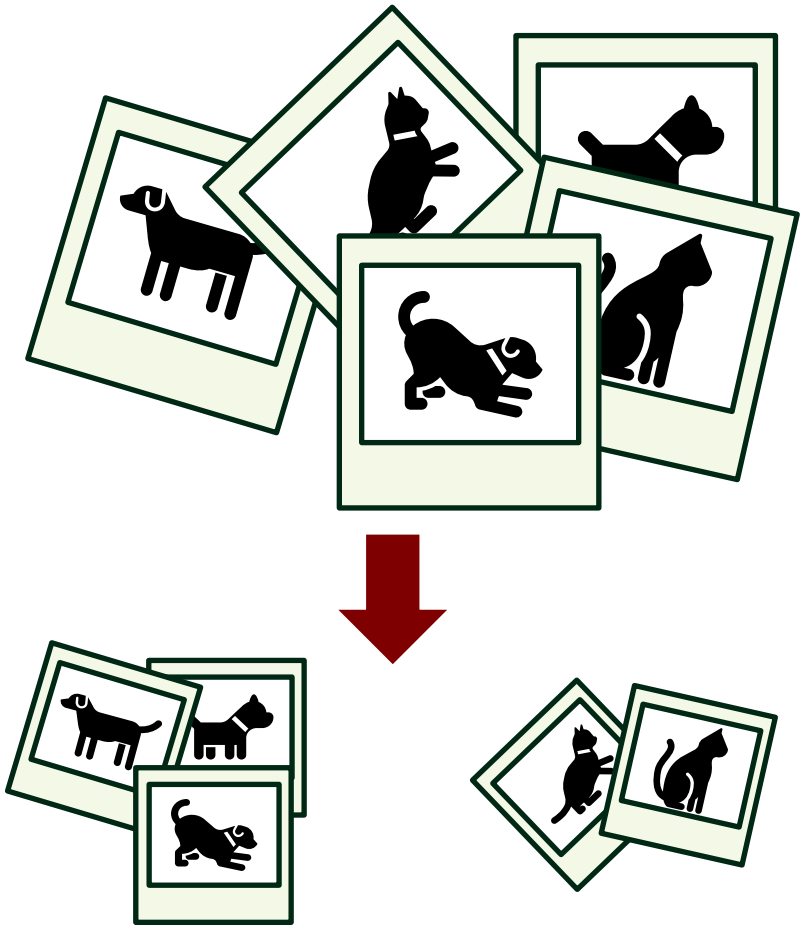


Image source: ISLP

Heuristic algorithm

Algorithm 12.2 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

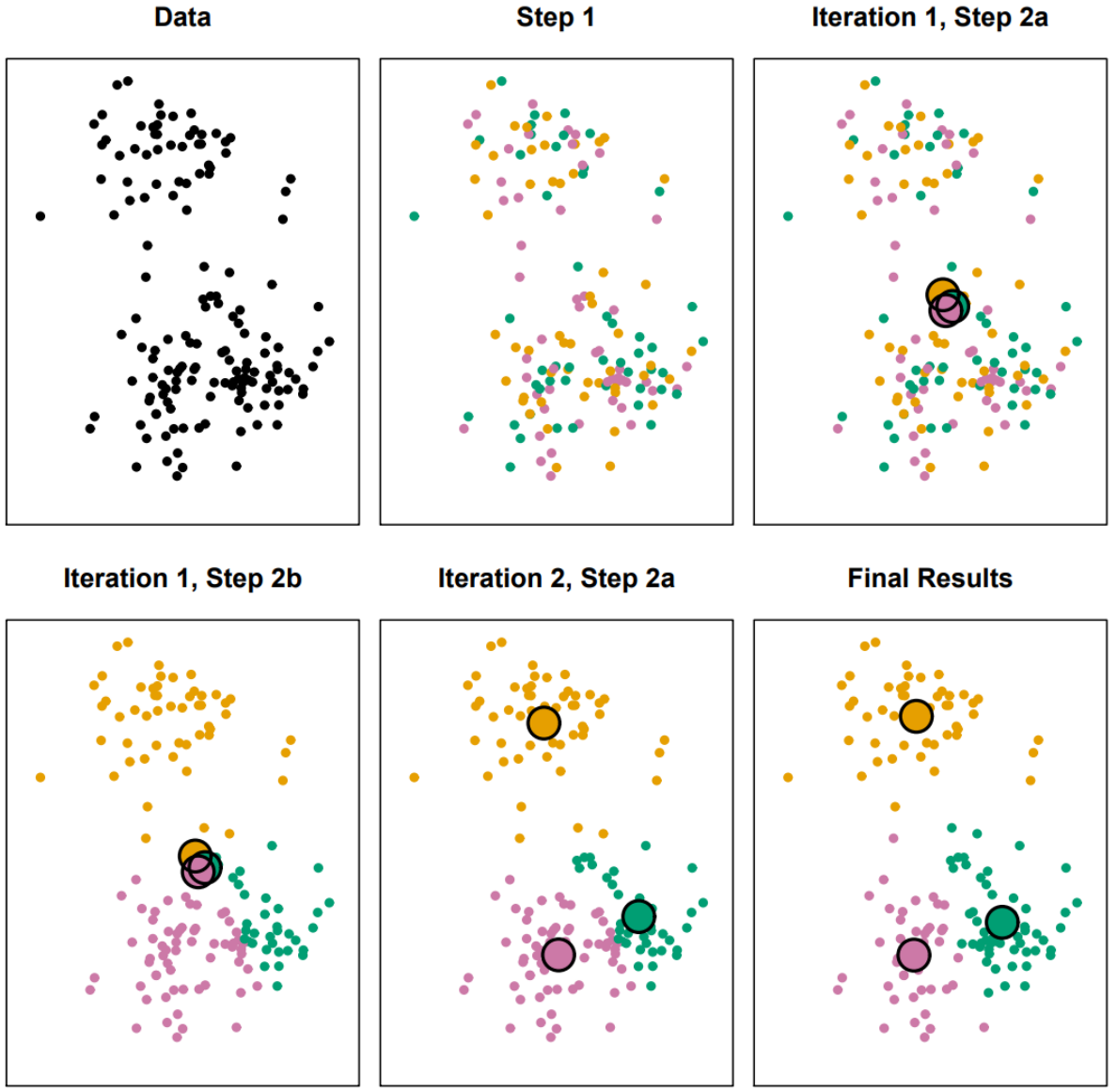


Image source: ISLP

Initialization

- Heuristic k-means algorithm finds local minima
  - Solution depends on initialization
- Ensemble approach:
  - Run algorithm with several different initializations
  - Compare outcomes using some scoring metric
  - Select the best-scoring outcome

Image source: ISLP

Dependent on hyperparameters

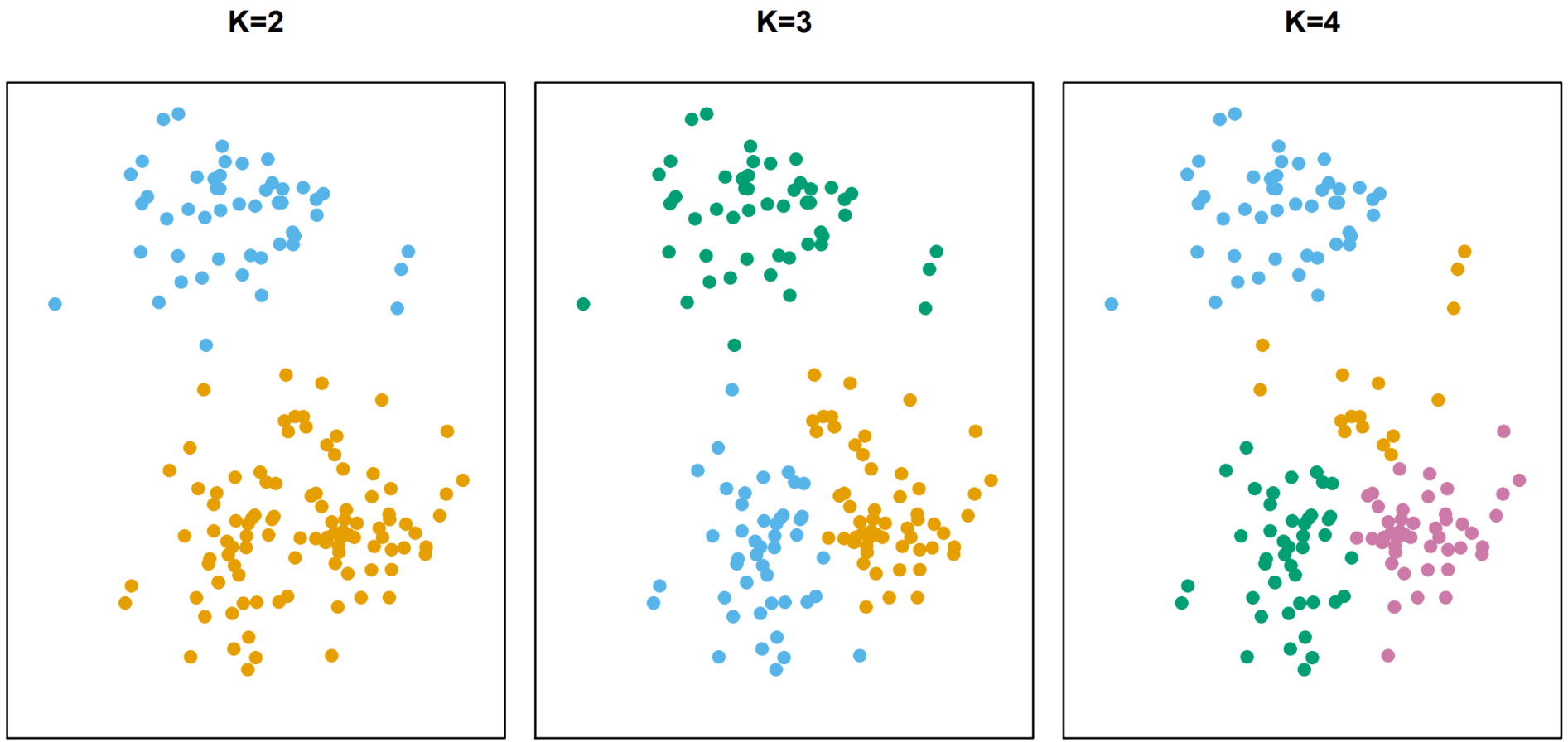


Image source: ISLP