CS 540 Introduction to Artificial Intelligence

Deep Learning IV

University of Wisconsin-Madison
Spring 2026 Sections 1 & 2

Midterm Information

- **Time: March 24th 5:45-7:15 PM**
- **Location (by section **):**
  - Section 001 (Tuesday/Thursday 11-12:15PM): 6210 Social Sciences Bldg
  - Section002 (Tuesday/Thursday 2:30-3:45PM): B10 Ingraham Hall
  - Students with McBurney accommodations should have received an email with additional information.
  - Students who cannot take the exam on the specified time should contact their instructor if they have not done it yet.
- **Topics: Topics covered up to and including Week 9**
- **Exclusion List (questions regarding the following topics will NOT appear on the midterm):**
  - **Logic (covered in sections 1 and 2)**
  - **SVM + Kernel Trick (covered in section 3)**
- **Format**: MCQ
- **Cheat sheet**: a handwritten single piece of paper, front and back
- **Calculator**: optional, if it doesn't have an Internet connection
- **Bring**: your WISC ID, pencil (No 2 or softer), your 1-sheet notes.
- **Past exam questions**: on Canvas →Files → Past Exams

Announcements

- **Homework 7 released and due Wednesday April 8 at 11:59PM.**

- **Additional optional review session on Friday March 20th at 11:00 AM on ZOOM (link provided on Canvas)**
- TA Review session **today at 5:30PM MH3610**
- Additional Office hours: **3:45-5 PM Monday March 23rd**

- Class roadmap and schedule:

Machine Learning: Deep Learning III
ML/DL Summary

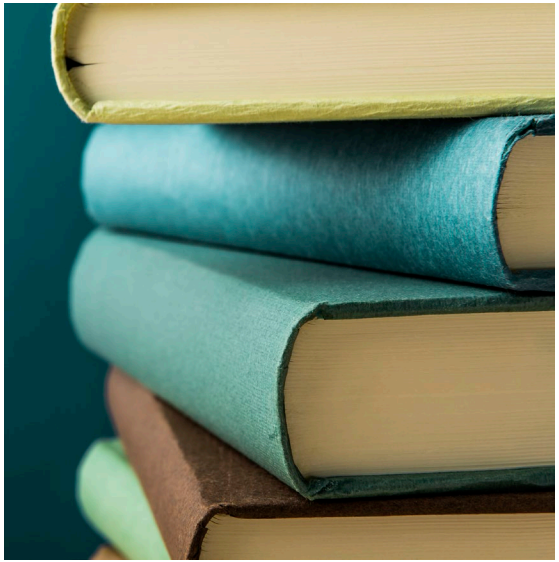
Outline

- Attention and Transformers
- Data Augmentation & Regularization
  - Expanding the dataset, avoiding overfitting
- More Signal From our Data
  - Graph-structured data, graph neural networks

The Attention Mechanism

From RNNS to Transformers

- RNNs process one word at a time — can't parallelize (Slow on GPUs.)
- Even LSTMs struggle with very long sequences
- All context gets squeezed into one hidden state vector

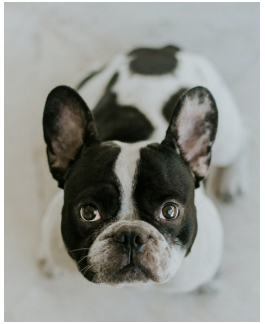


From RNNS to Transformers

- Transformers allow parallelization and efficient context handling
- Example: “The cat the dog chased ran away.”
  - Who ran away?
  - Need to remember/attend to earlier words
- Goal: **decide which words matter most!**

Word Representations & Context

- We use vectors to represent words (“embeddings”)
- Recall:
  - One-hot representation “dog”
    $[0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0]$
  - Dense embedding
    - Vector captures **meaning** $[0.13 \quad 0.87 \quad -0.23 \quad 0.46]$



Problem: the meaning of a word depends on the words around it

Word Representations & Context

"The monkey ate the banana. It was ____, wasn't it?"

Does "it" mean **monkey** or **banana**?

Could be:

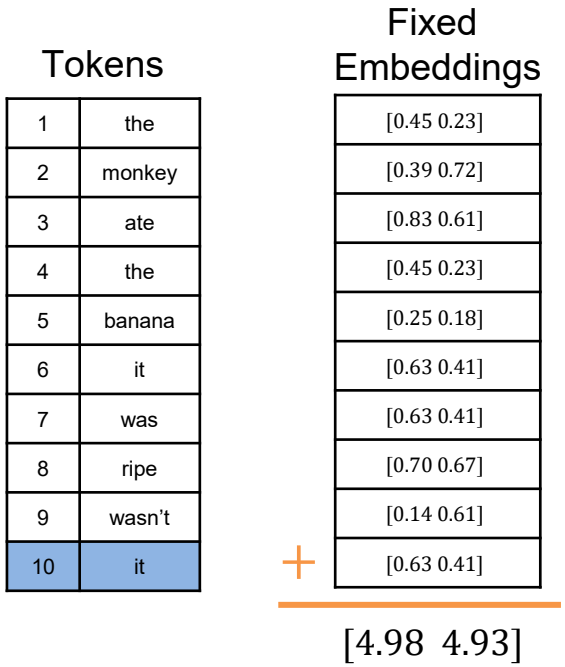
- The monkey ate the banana. It was ripe, wasn't it?
- The monkey ate the banana. It was hungry, wasn't it?

Meaning depends on past words (**context**)

The attention mechanism produces **contextual embeddings.**

Attempt 1: Naïve Contextual Embedding

- Each token has a fixed embedding vector x_i
- A crude attempt at contextual embedding: average over context
- Equal "attention" to every previous token



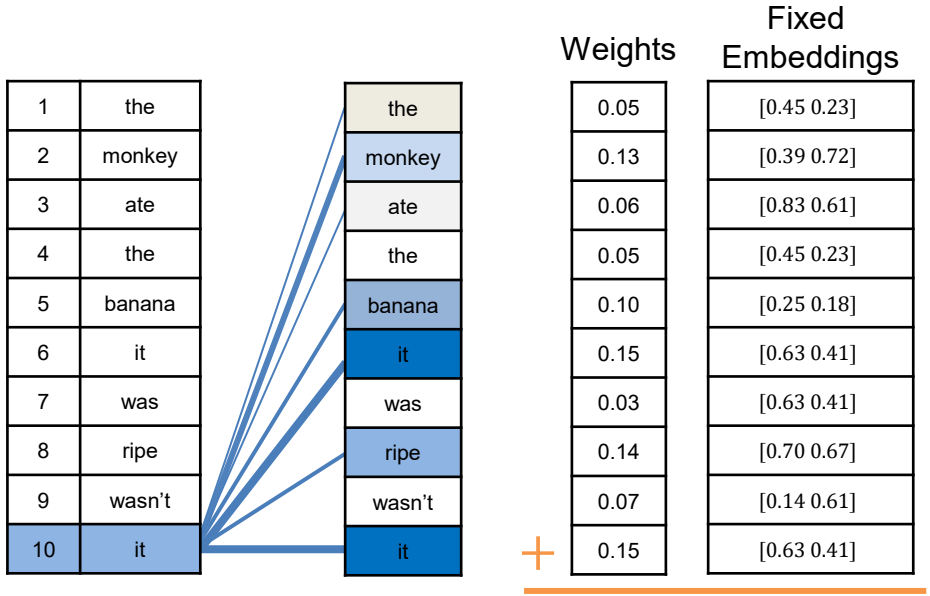
In math:
for the i-th token

$$c_i = \frac{1}{i}\sum_{j=1}^{i} x_j$$

Contextual embedding for "it": $[0.498 \;\; 0.493]$

Attempt 2: Assigning Weights

- Humans focus selectively — machines can too
- We can assign weights based on relevance
  - Idea: weight similar words highly
  - If $\langle x_i, x_j \rangle$ large, assign large weight
- Then take weighted sum



Contextual embedding for "it": $[0.37 \;\; 0.42]$

In math: for i-th token

$$r_{ij} = \frac{1}{\sqrt{d}}\langle x_i, x_j \rangle \qquad p_{i,:} = \mathrm{softmax}(r_{i,:}) \qquad c_i = \sum_{j=1}^{i} p_{ij} \cdot x_j$$

Final Attempt: The Attention Mechanism

Previous attempt:

- Fixed embeddings in three locations.

$$r_{ij} = \frac{1}{\sqrt{d}} \cdot \langle x_i, x_j \rangle$$

$$p_{i,:} = \mathrm{softmax}(r_{i,:})$$

$$c_i = \sum_{j=1}^{i} p_{ij} \cdot x_j$$

In the **attention mechanism**:

- Each token is associated with **three** vectors
- Query: q_i, the one attended from
- Key: k_j, the one attended to
- Value: v_j, the context being generated

Query Key

$$r_{ij} = \frac{1}{\sqrt{d}} \cdot \langle q_i, k_j \rangle$$

$$p_{i,:} = \mathrm{softmax}(r_{i,:})$$

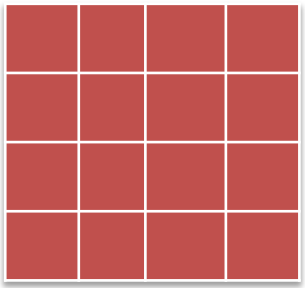
Value

$$c_i = \sum_{j=1}^{i} p_{ij} \cdot v_j$$

Query, Key, and Value Matrices

Input vector

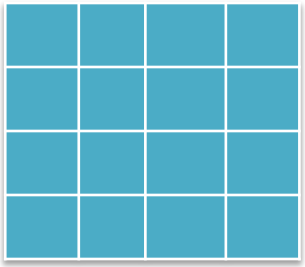
x



W_q

$q = W_q x$

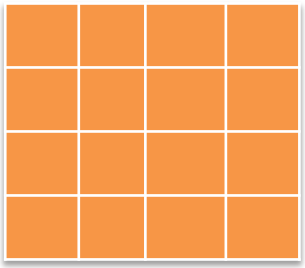
Query



W_k

$k = W_k x$

Key



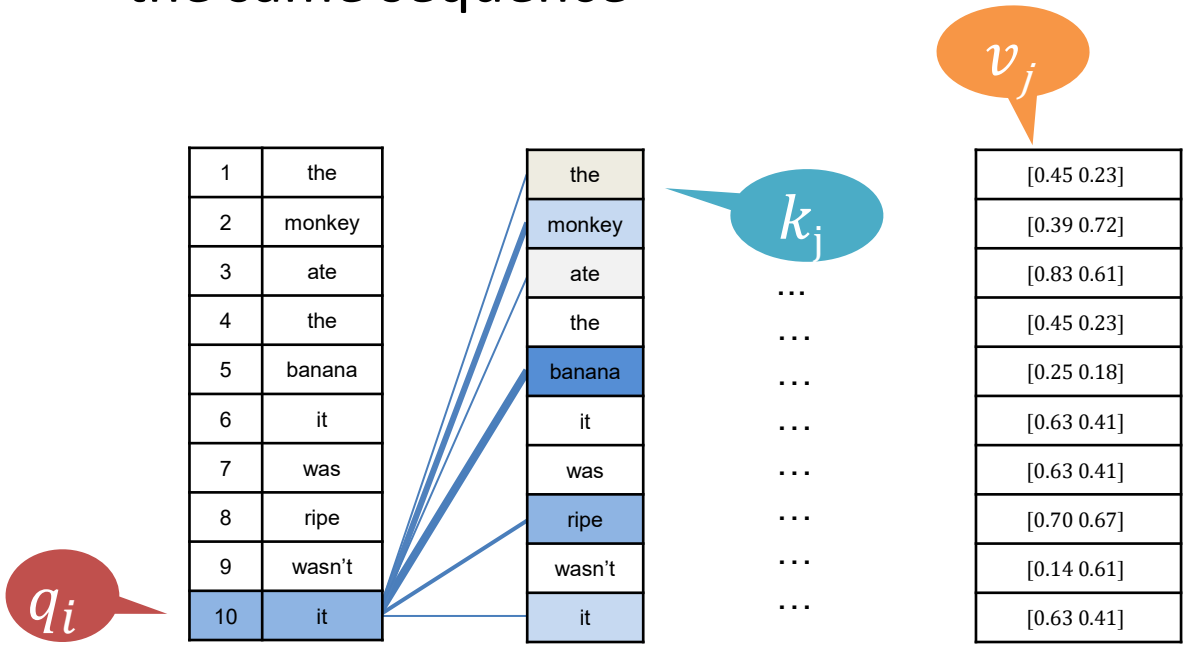
W_v

$v = W_v\ x$

Value

The Attention Mechanism

Each token attends to all other tokens in the same sequence



$$r_{ij} = \frac{\langle q_i, k_j \rangle}{\sqrt{d}}$$

$$p_{i,:} = \mathrm{softmax}(r_{i,:})$$

$$c_i = \sum_j p_{ij} \cdot v_j$$

Notation for Attention

Queries, keys and values are written as matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$

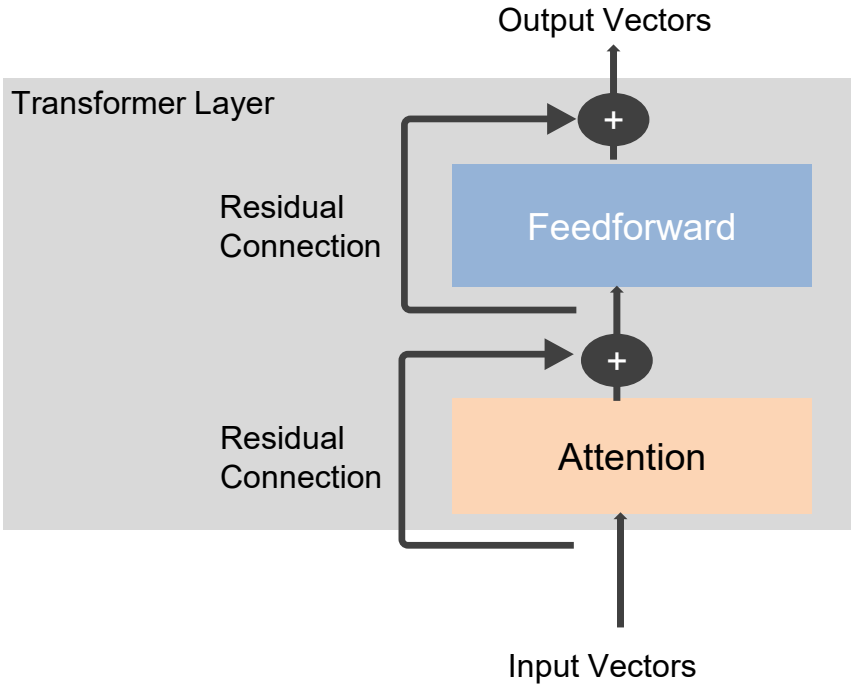
$$\text{Attention}\ (Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)\text{V}$$

The Transformer Architecture

From Attention to Transformer

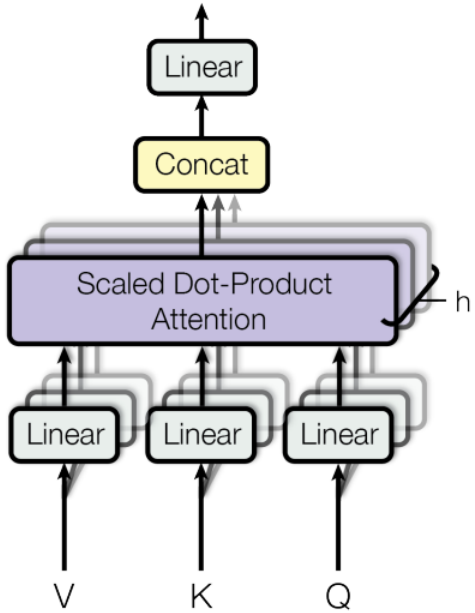
A single layer transformer consists of:

- Attention Mechanism
- Feed-Forward Network
- Residual Connections



Multi-Head Attention

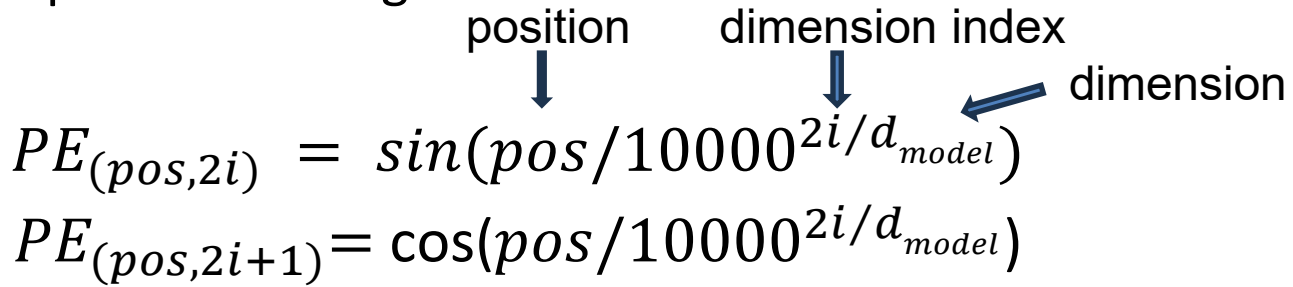
- Outputs combined for richer representations
- Multiple heads learn different relationships (syntax, meaning, position)



Vaswani, A., et al. (2017). Attention Is All You Need

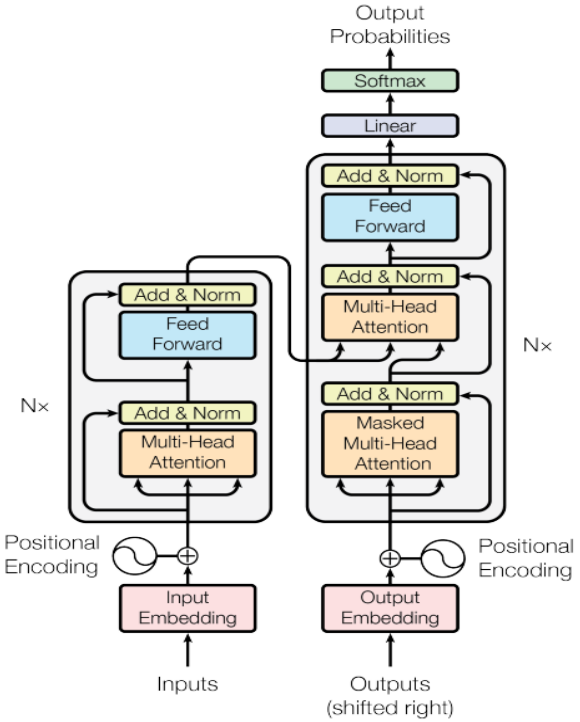
Positional Encoding

- Transformers have no recurrence — order must be added explicitly
- **Positional Encoding**: Information about the relative or absolute position of the tokens in the sequence
- Added to the input embeddings



Transformer Architecture

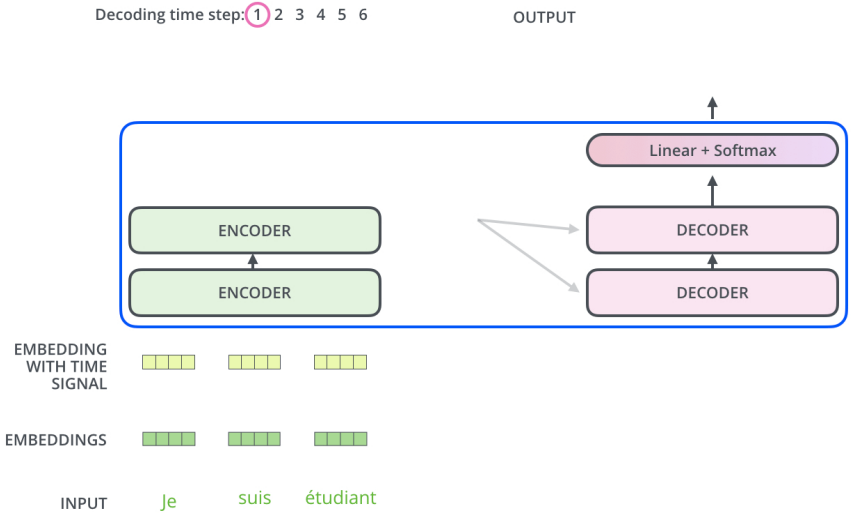
- Encoder–Decoder structure
- **Encoder**: maps an input sequence to a sequence of continuous representations z.
  - Useful for classification
- **Decoder**: Given z, the decoder generates an output sequence of symbols one element at a time.
  - Useful for generation



Vaswani, A., et al. (2017). Attention Is All You Need

Decoder

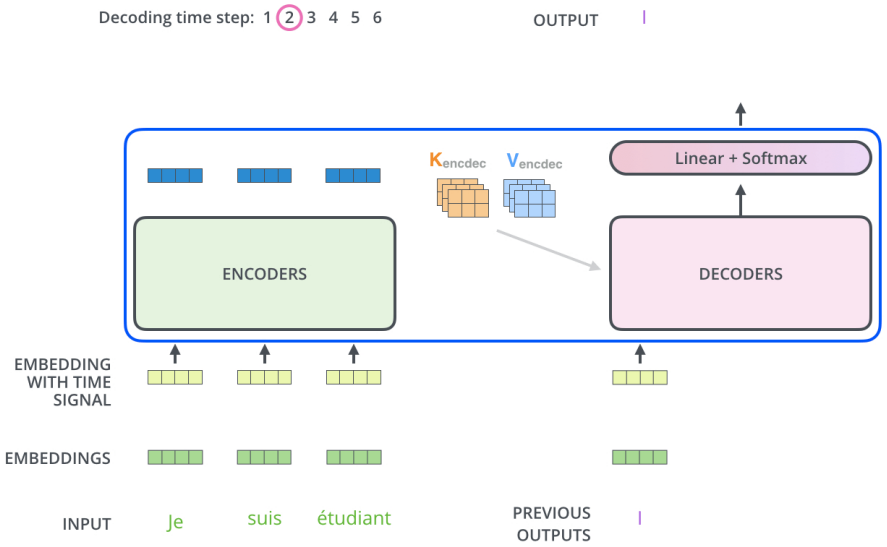
- **Masked multi-head attention**: each word attends to the words before it
- A **second attention** module that **attends the output of the encoder**



The Illustrated Transformer

Decoder

- **Masked multi-head attention**: each word attends to the words before it
- A **second attention** module that **attends the output of the encoder**



The Illustrated Transformer

Q3.1 Quiz Break

What is the primary function of the self-attention mechanism?

A) To track the position of each word in the sequence.
B) To process the input sequence strictly from left to right.
C) To weigh the importance and relationship of all words in a sequence relative to each other.
D) To reduce the size of the model by using fewer parameters.

Q3.1 Quiz Break

What is the primary function of the self-attention mechanism?

A) To track the position of each word in the sequence.
B) To process the input sequence strictly from left to right.
C) To weigh the importance and relationship of all words in a sequence relative to each other.
D) To reduce the size of the model by using fewer parameters.

Q3.2 Quiz Break

In the context of the Transformer model, what role do "Encoders" and "Decoders" play?

A) Encoders are used for text generation, and Decoders are used for text classification.
B) Encoders process the input sequence, and Decoders generate the output sequence.
C) Both Encoders and Decoders are only used for understanding the input sequence.
D) Both Encoders and Decoders map an input sequence to a sequence of continuous representations.

Q3.2 Quiz Break

In the context of the Transformer model, what role do "Encoders" and "Decoders" play?

A) Encoders are used for text generation, and Decoders are used for text classification.

B) Encoders process the input sequence, and Decoders generate the output sequence.

C) Both Encoders and Decoders are only used for understanding the input sequence.

D) Both Encoders and Decoders map an input sequence to a sequence of continuous representations.

Applications

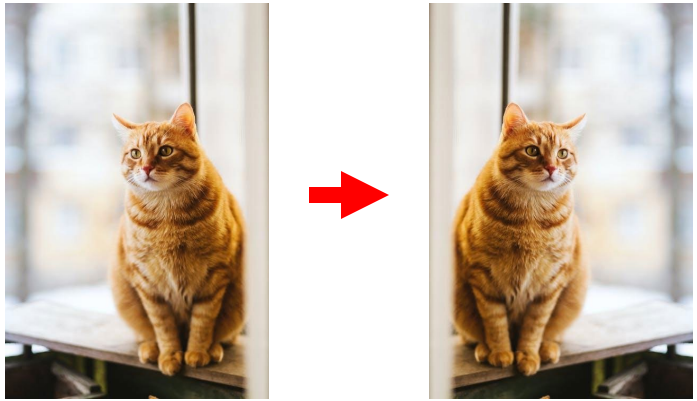
- Language Models: GPT, BERT, T5
- Vision: ViT (Vision Transformer)
- Multimodal: CLIP, DALL·E, GPT-4v
- Scientific: AlphaFold, time-series modeling, robotics

Data Concerns

Data Concerns

What if we don’t have a lot of data?

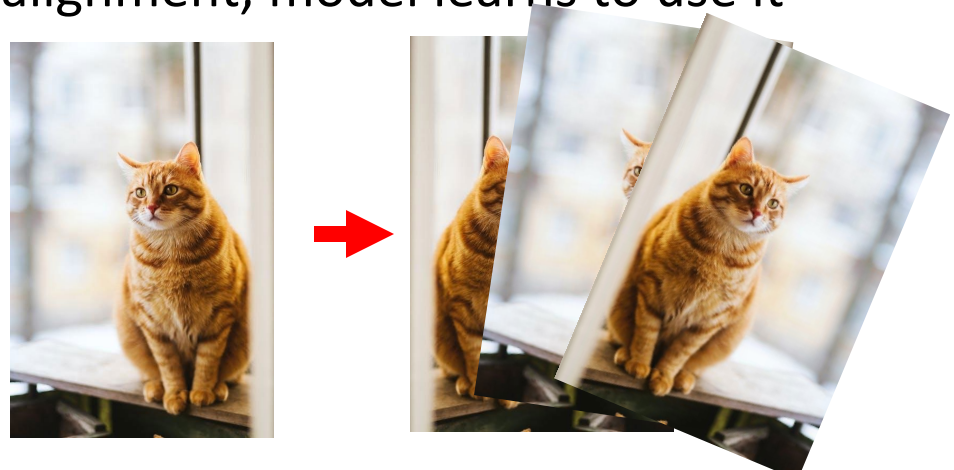
- We risk overfitting
- Avoiding overfitting: **regularization** methods
- Data augmentation: a classic way to regularize



Data Augmentation

Augmentation: transform + add new samples to the training set

- Transformations: based on domain
- Idea: build **invariances** into the model
  - **Ex**: if all images have same alignment, model learns to use it
- Keep the label the same!



Transformations

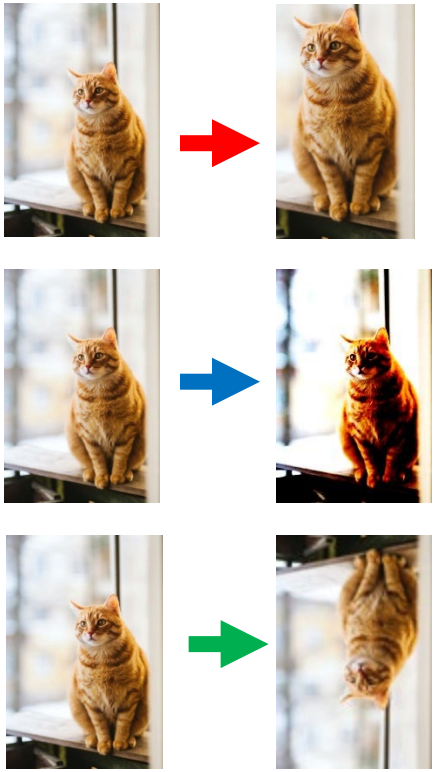
Examples of transformations for images

- **Crop** (and zoom)
- **Color** (change contrast/brightness)
- **Rotations+** (translate, stretch, shear, etc)

Many more possibilities. Combine as well!

Q: how to deal with this at **test time**?

- A: transform, test, average



Combining & Automating Transformations

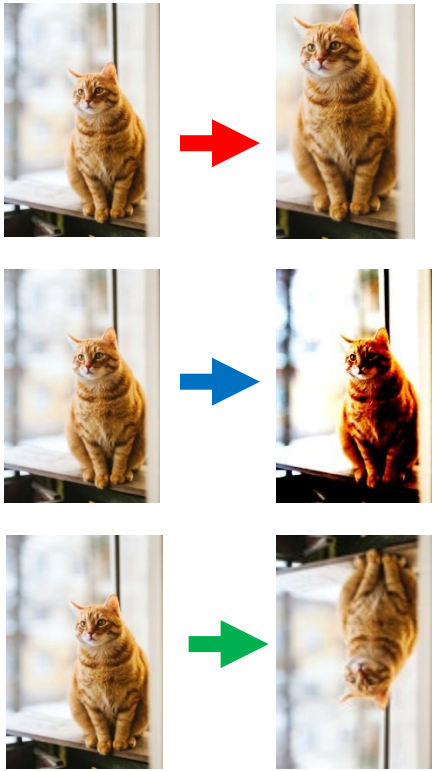
One way to automate the process:

- Apply every transformation and combinations
- **Downside:** most don't help…

Want a good policy, ie, → → → → →

- Active area of research: search for good policies

1. **Ratner et al**: "Learning to Compose Domain-Specific Transformations for Data Augmentation"
2. **Cubuk et al**: "AutoAugment: Learning Augmentation Strategies from Data"



Other Domains

Not just for image data. For example, on text:

- Substitution
  - E.g., “It is a **great** day” ➔ “It is a **wonderful** day”
  - Use a thesaurus for particular words
  - Or, use a model. Pre-trained word embeddings, language models
- Back-translation
  - “Given the low budget and production limitations, this movie is very good.” ➔ “There are few budget items and production limitations to make this film a really good one”

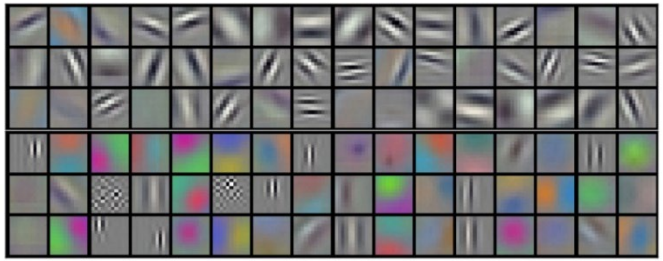
Xie **et al**: “Unsupervised Data Augmentation for Consistency Training”

Importance of Augmentation

Data augmentation is critical for top performance!

- You should use it!
- **AlexNet**: used (many papers re-used as well)
  - Random crops, rotations, flips.

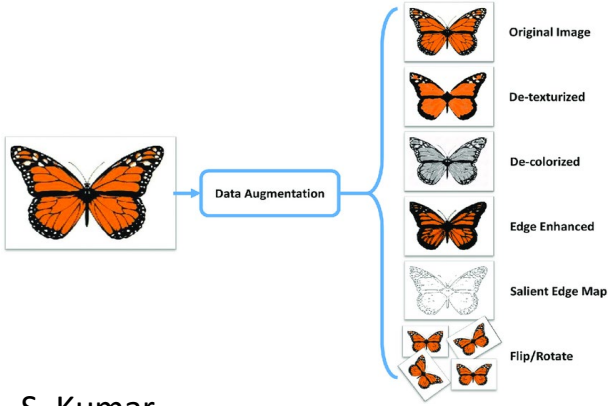
Krizhevsky et al: “ImageNet Classification with Deep Convolutional Neural Networks”



Other Forms of Regularization

Regularization has many interpretations

- **Goodfellow**: “any modification… to a learning algorithm that is intended to reduce its generalization error but not its training error.”
- A way of adding knowledge / side information to model
- Enforcing parsimony/simplicity



Other Forms of Regularization

Classic regularizations

1. Modify loss functions

   Ex: regularized least squares LR

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\theta_0 + x_i^T \theta - y_i)^2 + \lambda \|\theta\|_2^2$$

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f_\theta(x_i), y_i) + \lambda R(f_\theta)$$

Regularizer

Standard loss

Regularization parameter

2. Modify architecture/training/data

   a) Dropout, batch normalization, augmentation

Break & Quiz

Q 2.1: If we apply data augmentation blindly, we might

(i) Change the label of the data point

(ii) Produce a useless training point

- A. (i) but not (ii)
- B. (ii) but not (i)
- C. Neither
- D. Both

Break & Quiz

Q 2.1: If we apply data augmentation blindly, we might

(i) Change the label of the data point

(ii) Produce a useless training point

A. (i) but not (ii)

B. (ii) but not (i)

C. Neither

D. Both

Break & Quiz

Q 2.1: If we apply data augmentation blindly, we might

(i) Change the label of the data point

(ii) Produce a useless training point

A. (i) but not (ii) (Can do (ii): imagine turning up the contrast till the image is completely black and is unusable).

B. (ii) but not (i) (Can change label: rotate a 6 into a 9).

C. Neither (Can do either).

D. Both

Break & Quiz

Q 2.2: Which of the following is false about regularization?

A. Data augmentation can be applied for text.

B. Regularization always introduces complexity to a model to avoid overfitting.

C. We can modify the loss function by adding a regularization term.

D. The transformations for data augmentation depend on the domain.

Break & Quiz

Q 2.2: Which if the following is **false** about regularization?

A. Data augmentation can be applied for text.

B. Regularization always introduces complexity to a model to avoid overfitting.

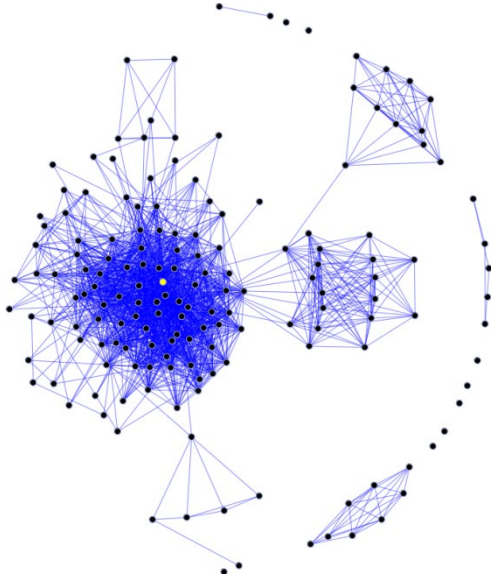
C. We can modify the loss function by adding a regularization term.

D. The transformations for data augmentation depend on the domain.

Graph Neural Networks

Relationships in Data

- We usually assume all data points are independent, “unrelated” in a sense
- Pretty common to have relationships between points $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$
  - **Social networks**: individuals related by friendship
  - **Biology/chemistry**: bonds between compounds, molecules
  - **Citation networks**: Scientific papers cite each other

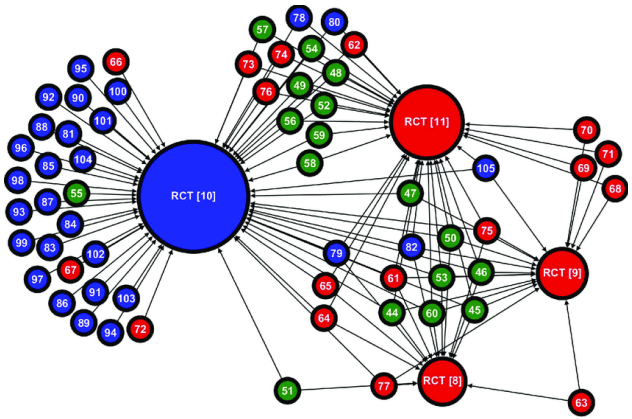


Wiki

Signal from Relationships

Suppose we are classifying scientific papers

- **Features**: title, abstract, authors. **Labels**: math/science/eng.
- Could build a reasonable classifier with the above data
- **More signal** from relationships
  - Cite each other, more likely from the same field
  - Note: citations are not features; they're **links**
  - Need a new type of network to handle



Leng

Graph Neural Networks

Have: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n), G = (V, E)$

How should our new architecture look?

- Still want layers
  - linear transformation + non-linearity

Parameters

Non-Linearity

$$H^{(\ell+1)} = \sigma(H^{(\ell)} W^{(\ell)})$$

Hidden Layer Representation

- Now want to integrate neighbors
- Bottom: graph convolutional network

$$H^{(\ell+1)} = \sigma(A_G H^{(\ell)} W^{(\ell)})$$

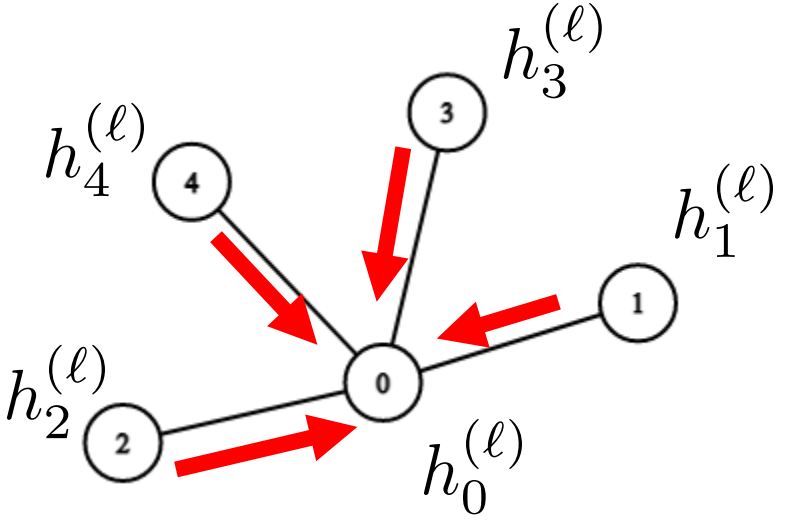
Graph Mixing

Kipf and Welling: “Semi-Supervised Classification with Graph Convolutional Networks”

Graph Convolutional Networks

Let's examine the GCN architecture in more detail

- Difference: "graph mixing" component
- At each layer, get representation at each node
- Combine node's representation with neighboring nodes

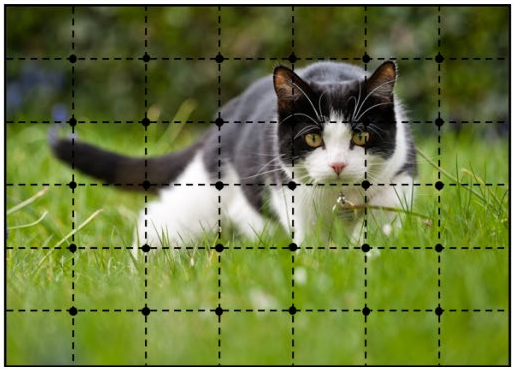


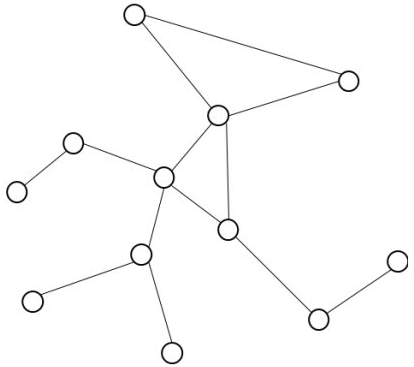
- "**Aggregate**" and "**Update**" rules

Graph Convolutional Networks

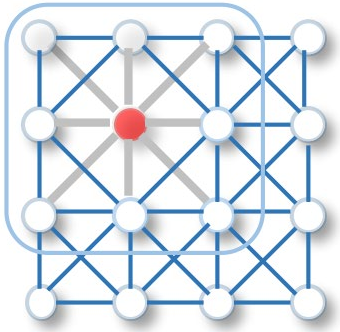
Note the resemblance to CNNs:

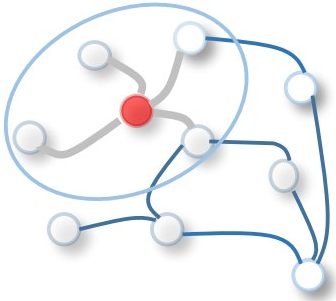
- Pixels: arranged as a very regular graph
- Want: more general configurations (less regular)





Wu et al, A Comprehensive Survey on Graph Neural Networks





Zhou et al, Graph Neural Networks: A Review of Methods and Applications

Further Reading/Viewing

- Jurafsky & Martin, Chapter 8
  - https://web.stanford.edu/~jurafsky/slp3/ed3book_aug25.pdf
- Russell & Norvig, Chapter 21.6 and 24

- Andrej Karpathy tutorial
  - https://karpathy.ai/zero-to-hero.html
- 3Blue1Brown:
  - https://www.youtube.com/watch?v=eMlx5fFNoYc
- The Illustrated Transformer
  - https://jalammar.github.io/illustrated-transformer/