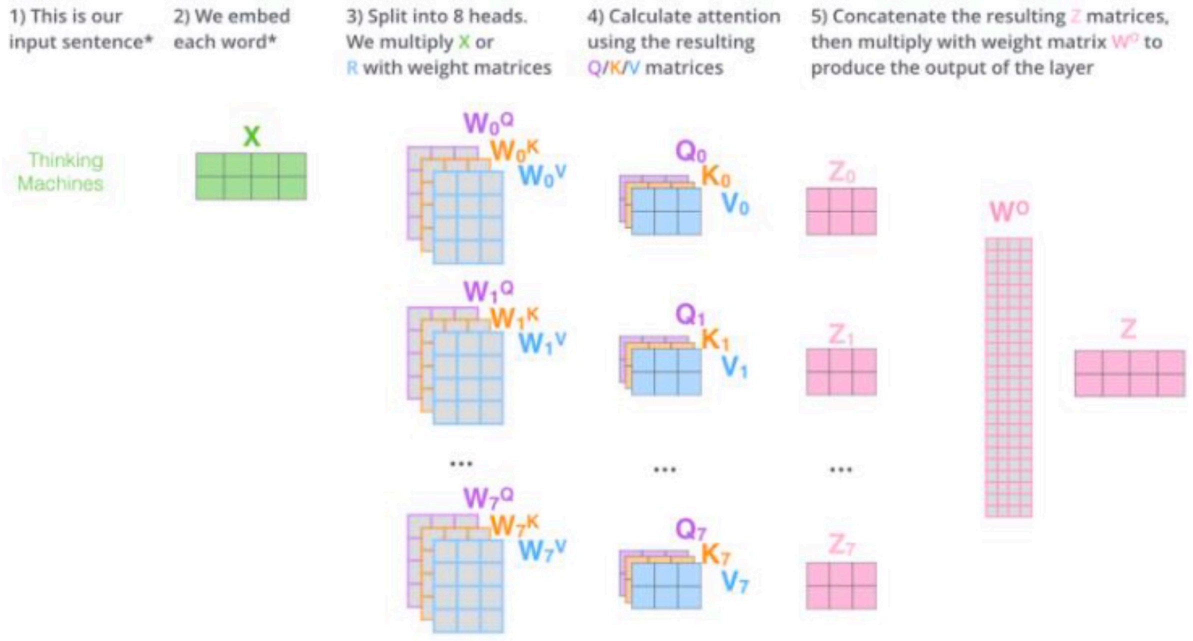
Organizational

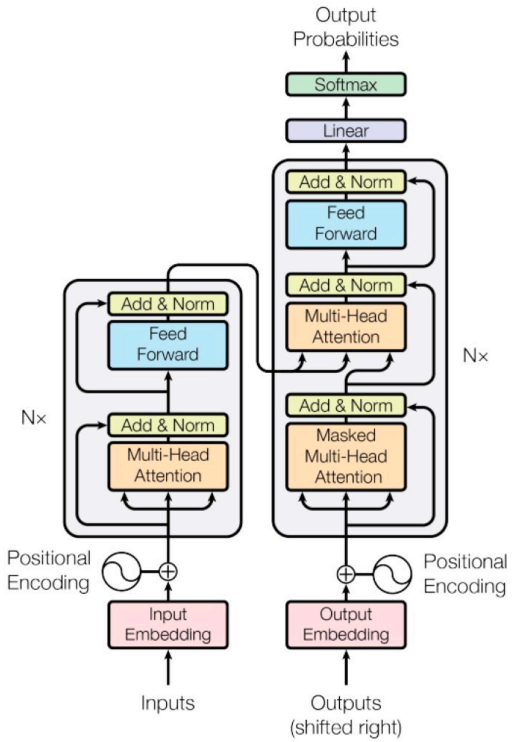
- Homework: **You nearly did it!**
  - Homework 7 released (you will have until Dec 6^{th})
  - → No more extensions
- Midterm Wednesday 11/29 4.30 to 5.45

Transformers

(Image credit: Paramount Pictures)

Transformers





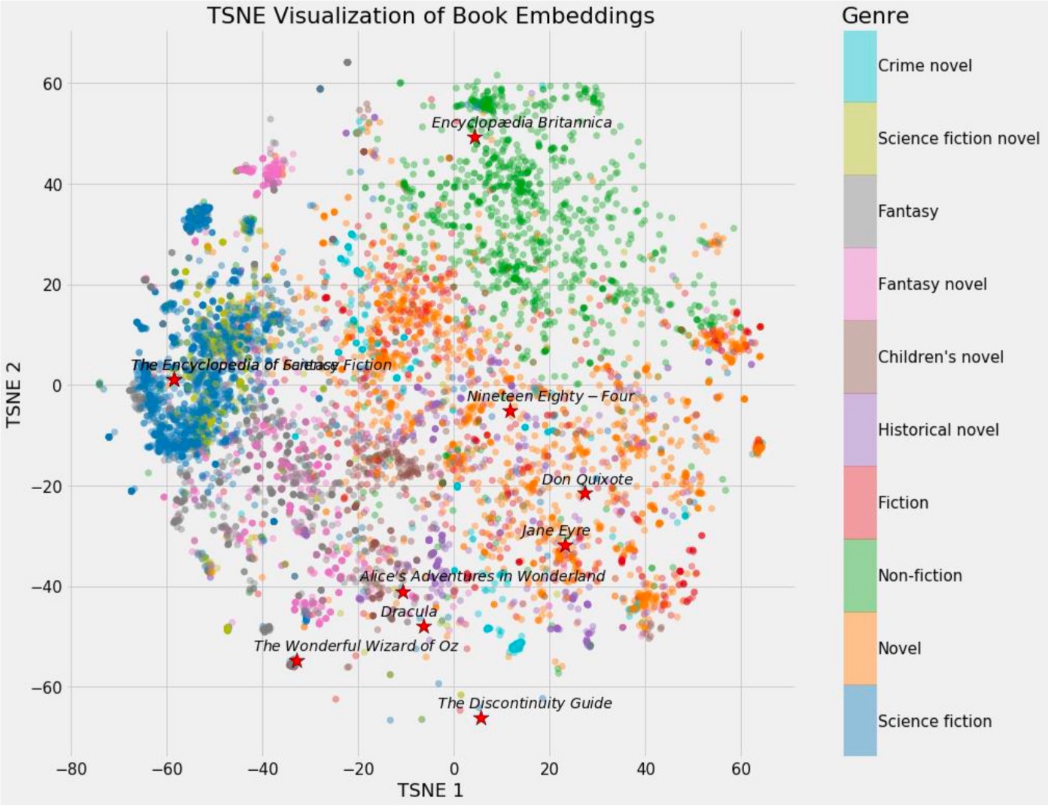
EN.601.482/682 Deep Learning Mathias Unberath

Embedding

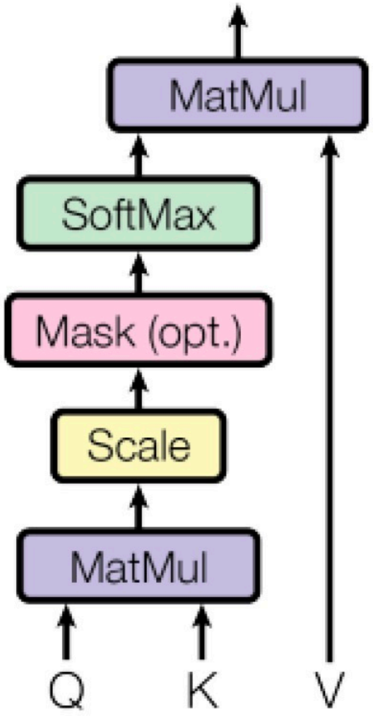
- From one-hot encoding (image patches) to a continues space.

- Dimensionality reduction.

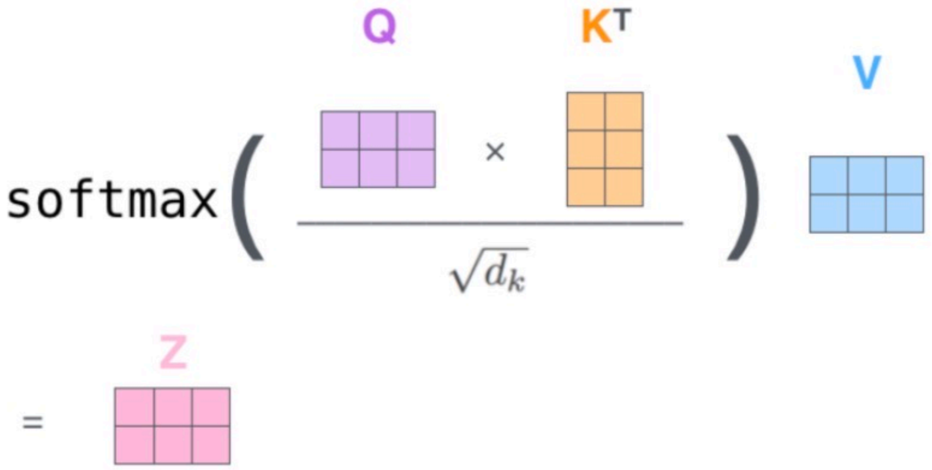
- Semantic Clusters



Attention



*attention (q, k, v) = Softmax (q * k^T) * v*

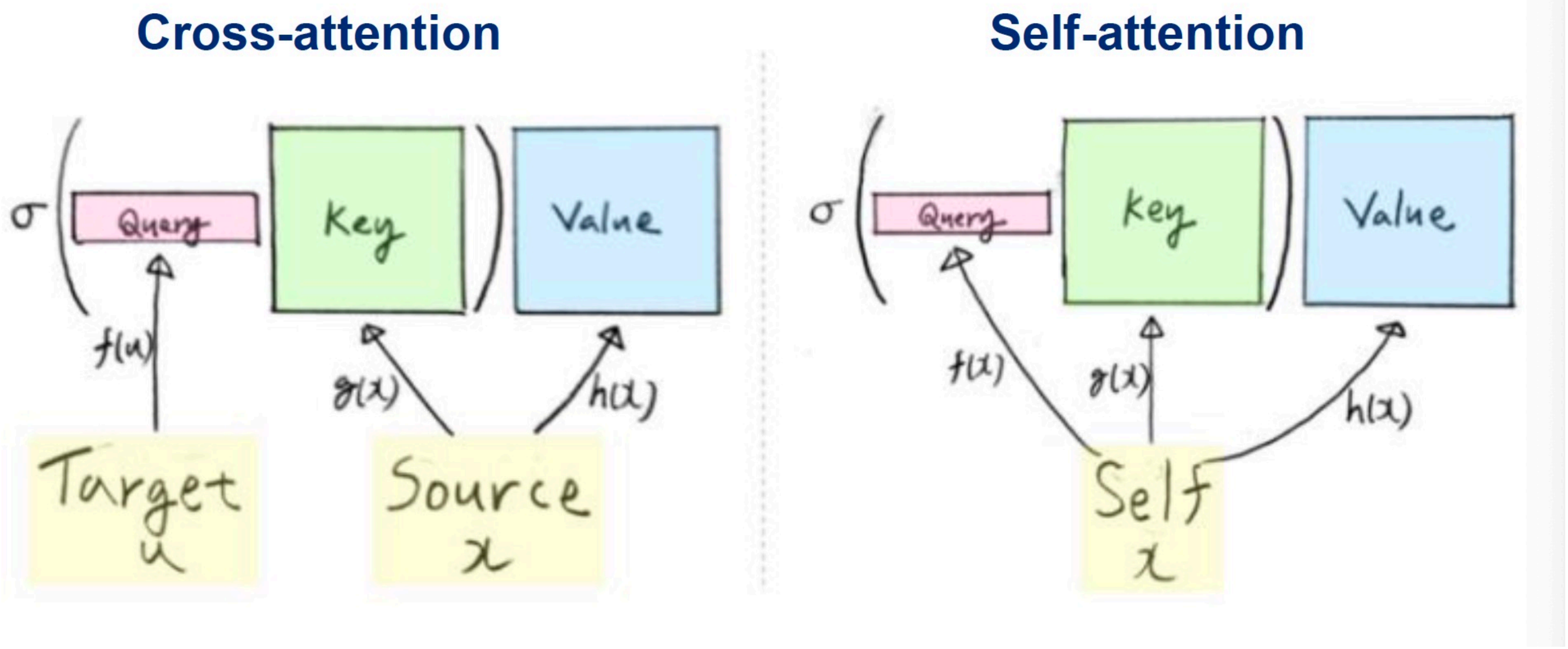


What exactly is Attention?

- Feature aggregation for sequence.
- How do RNN do feature aggregation? -> sum them up.
- How do attention do feature aggregation? -> weighted sum.

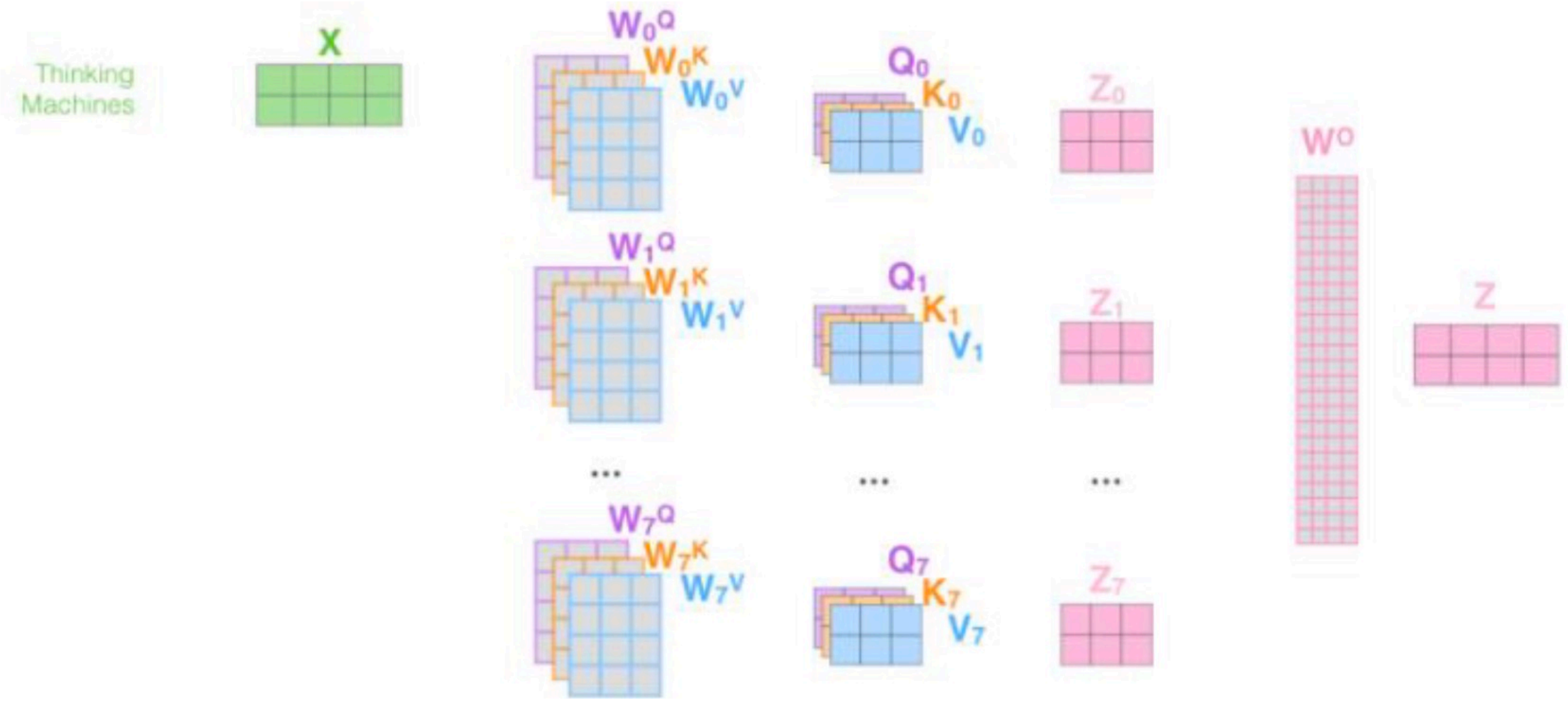
$$attention\ (q, k, v) = \underbrace{\boxed{\underbrace{\boxed{Softmax\ (q * k^T)}}_{\text{Weight}} * v}}_{\text{Weighted Sum}}$$

Several Attentions

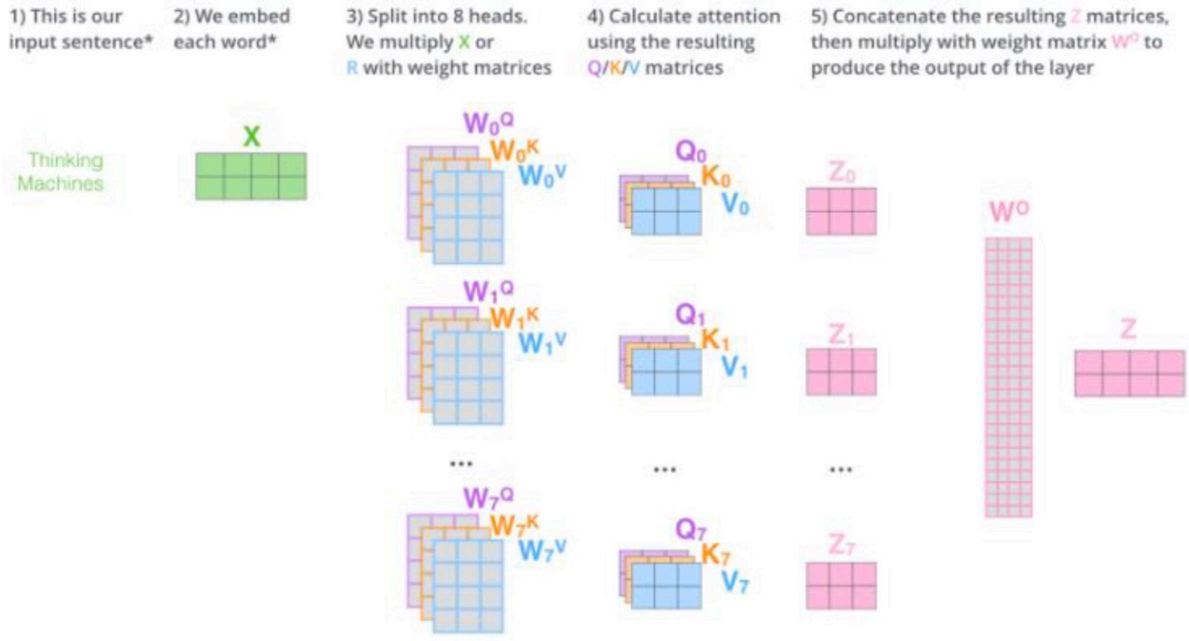


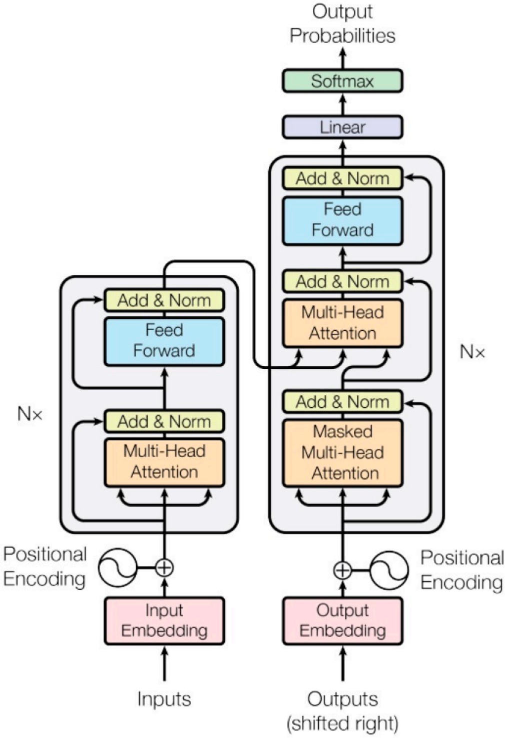
Several Attentions

Multi-head Attention



Transformers





EN.601.482/682 Deep Learning Mathias Unberath

Transformer vs RNN

- Transformer can do parallel training.
- Transformer keeps the form of the whole sequence.

Challenges that the Transformers are facing

- Broken Cybertron
- O(n^2) complexity for attention.
- O(n^2) complexity during inference-time.

- Take look at Retentive Network if you are interested in this.

HW7

- Three **bonus** problems: +1% towards hw per problem
- Released, check Piazza
- Due: Wednesday, Dec 6th by 11:59 pm
- Submit: a zip to Gradescope (Entry Code **BBVDNN**)

Q1. Generative Adversarial Networks

Training a GAN to generate MNIST image

1. Define a generator
2. Define a discriminator
3. Minimax optimization
4. Goal: generate reasonable image of inception score >= 1.5

Q2. Adversarial Attacks

1. Implement a gradient-based attack method
2. Generate adversarial example for MNIST
3. Generate adversarial example for CIFAR-10
4. Generate adversarial example for a CIFAR-alike real world photo

More fun (optional):

Play with a physical attack method on an object detection algorithm!

Q3. Word Embedding

1. Implement two word embedding methods:
   a. CBOW (Continuous Bag of Words)
   b. Skip-gram
2. Train the model to generate good embeddings
3. Use the embeddings to evaluate words similarity

Balancing your efforts on

bonus problems (<=3% of 50%)

vs.

the final project (25%)