

# Attention

---

Arush Iyer

# High-Level

- Lets models to selectively focus on the most important parts of input data
  - Input data: keywords of a sentence, pixels of an image, etc.
  - In our problem: tells us which regions of the tissue slice are most useful to look at
  - Originally designed for NLP, has applications in image classification
- Assigns weights to each input feature
  - Tells the model which features are the most important
- Why should we care?
  - Interpretability: Allows us to visualize what the model is looking at when making decisions
  - Efficiency: Model learns to look at important areas and ignore irrelevant information
- Potential downsides:
  - Increased complexity
  - Can struggle with noisy and/or small datasets
- Applications in our code
  - Attention mechanisms for CNN architectures (CBAM, SE-Resnet)
  - Post-hoc explanation methods (Grad-CAM)

# Types of attention

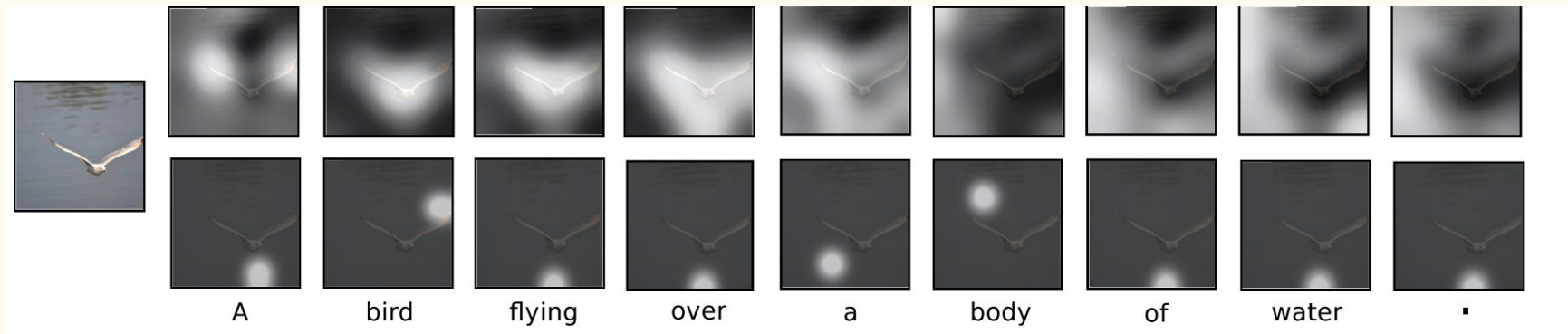
- Self attention mechanisms
  - Lets each input element consider its relationship with all other input elements
  - E.g., eyes and ears of a cat are spatially distinct but contextually related
  - Model: ViTs (Vision Transformers)
- Channel attention
  - Determines what feature channels (texture, colors, shapes, etc.) are important
  - “Should I pay more attention to the colors or the shapes in this picture?”
  - Model: SENet
- Spatial attention
  - Highlights which regions of a feature map are the most important
  - E.g., focusing more on the trees and less on the sky in a landscape painting
  - Models: CBAM and BAM (both combine Channel and Spatial but use different architectures)
- Post-hoc attention methods
  - Don't directly affect training but provide visual explanations for what the CNN did
  - Examples: Grad-CAM, CAM, Guided Backpropagation

# Soft vs hard attention

- Soft
  - Deterministic attention mechanism
  - Computes a weighted sum of inputs
    - Should all sum up to one
- Hard
  - Select just a few elements to look at
  - Stochastic attention mechanism
  - Typically modelled as sampling from a probability distribution
- Soft attention used in every popular attention method
  - Ease of training and generally better performance

# Soft vs hard attention visualized

Soft (top row) vs Hard (bottom row) attention



# Mathematical Foundation

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

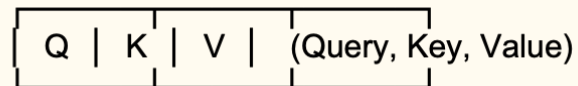
- Variables:
  - Q: Matrix of queries, or what the model is seeking
  - K: Matrix of Keys, or the available data points for the model
  - V: Matrix of values (e.g., color, shape, and texture of the patch)
  - $d_k, d_v$ : dimension of keys and values

# Mathematical Foundation

Input Feature Map



Linear Projection



Compute Similarity  
(Matrix Multiplication)

$$Q \times K^T$$



Scale and Normalize  
(Divide by  $\sqrt{d_k}$ , then Softmax)



Attention Map



Weighted Sum  
(Attention Map  $\times V$ )



Output Feature Map

# Applications: CBAM

- High-level:
  - Integrates spatial and channel attention through two sequential sub-models
  - First we figure out which features are important and then figure out where to look
- Math:
  - $\mathbf{M}_c(\mathbf{F})$ : channel attention map
  - $\mathbf{M}_s(\mathbf{F}')$ : spatial attention map
  - $\mathbf{F}$ : input feature map
  - $\mathbf{F}'$ : Channel refined features
  - $\mathbf{F}''$ : final refined features
  - $\otimes$ : element-wise multiplication

$$\begin{aligned}\mathbf{F}' &= \mathbf{M}_c(\mathbf{F}) \otimes \mathbf{F}, \\ \mathbf{F}'' &= \mathbf{M}_s(\mathbf{F}') \otimes \mathbf{F}',\end{aligned}$$



# Applications: CBAM

- Math (explained)
  - Channel submodule
    - Apply average and max pooling across each channel
    - Run those through an MLP
    - Apply a sigmoid activation function to get weights between 0 and 1

$$\mathbf{M}_c(\mathbf{F}) = \sigma(MLP(AvgPool(\mathbf{F})) + MLP(MaxPool(\mathbf{F})))$$









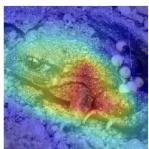

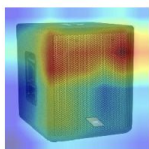
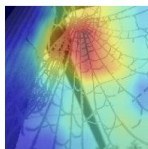

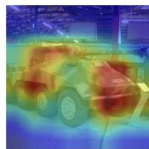
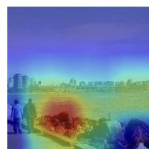
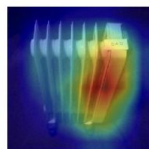
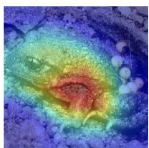

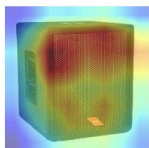
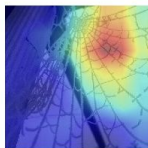

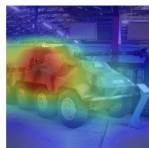

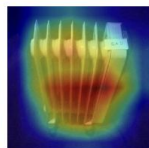
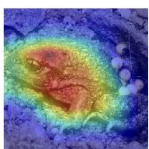

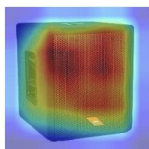
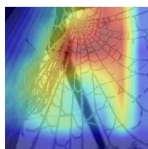

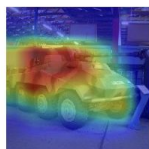

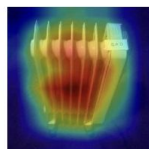
- Spatial submodule
  - Apply average and max pooling
  - Stack average and max maps together
  - Apply a convolutional layer (usually a 7x7 filter size)
  - Apply the sigmoid function

$$\mathbf{M}_s(\mathbf{F}) = \sigma(f^{7 \times 7}([AvgPool(\mathbf{F}); MaxPool(\mathbf{F})]))$$

# Applications: CBAM

- Pros
  - Guaranteed easier interpretability
  - Possible improved accuracy
- Cons
  - Very sensitive to hyperparameters
    - E.g., pooling methods, filter size
  - Potentially susceptible to noisy input data
    - Should be robust to this, still trying to figure out why

# Applications: CBAM

	Tailed frog	Toilet tissue	Loudspeaker	Spider web	American egret	Tank	Seawall	Space heater
Input image								
ResNet50								
	P=0.80736	P=0.11857	P=0.65681	P=0.22357	P=0.64185	P=0.14763	P=0.92236	P=0.01176
ResNet50 + SE								
	P=0.87240	P=0.14643	P=0.77550	P=0.25093	P=0.70827	P=0.15367	P=0.97166	P=0.26611
ResNet50 + CBAM								
	P = 0.96340	P = 0.19994	P = 0.93707	P = 0.35248	P = 0.87490	P = 0.53005	P = 0.99085	P = 0.59662

# SENet vs CBAM vs BAM

Feature	SENet (2017)	CBAM (2018)	BAM (2018)
Attention Types	Channel-only	Channel + Spatial (sequentially)	Channel + Spatial (in parallel, then combined)
Structure	Channel-wise global pooling + MLP	Channel attention followed by spatial attention	Parallel channel and spatial paths, then combined
Complexity	Least complex	Moderately complex	Moderately complex
Effectiveness	Effective for channels only	Generally more effective (spatial and channel)	Effective, similar to CBAM but different in implementation
Computational Cost	Lowest overhead	Moderate overhead	Moderate overhead
Flexibility	Less flexible (channel only)	Flexible (modular channel+spatial)	Flexible (parallel paths, jointly learned)

# Applications: Grad-CAM

- High-level:
  - Grad-CAM is a visualization technique to make CNN decisions interpretable
  - Works with almost all CNN architectures
- Steps:
  - Forward pass: pass image through CNN and get feature maps
  - Backward pass: get the gradients of the target class output with respect to activations from feature map
  - Global average pooling: average gradients to get weights (how important each channel is)
  - Make the heatmap and overlay it on original image

- Math:

$$L_{\text{Grad-CAM}}^c = \text{ReLU} \left( \underbrace{\sum_k \alpha_k^c A^k}_{\text{linear combination}} \right)$$

$$\alpha_k^c = \overbrace{\frac{1}{Z} \sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

# Applications: Grad-CAM

- Math (explained)
  - Calculating the importance weights  $\alpha_k^c$  for class  $c$  and channel  $k$ 
    - Finding gradient of the class score  $y^c$  with respect to the feature map  $A^k$ 
      - Feature map: what the model saw, usually extracted from last convolutional layer
      - Class score: probability score for each class before softmax
    - Global average pooling
      - Find average gradient value across entire feature map
      - Z: Number of positions in the map (essentially height \* width)
      - Large average gradient means that feature map was important for the particular class
  - Multiply each map by its score and then add them up

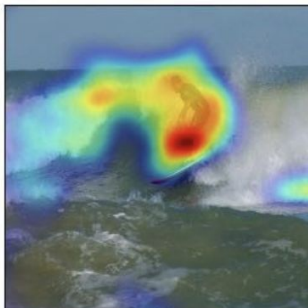
$$L_{\text{Grad-CAM}}^c = \text{ReLU} \left( \underbrace{\sum_k \alpha_k^c A^k}_{\text{linear combination}} \right)$$

$$\alpha_k^c = \overbrace{\frac{1}{Z} \sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

# Applications: Grad-CAM



What is the man doing?



Surfing



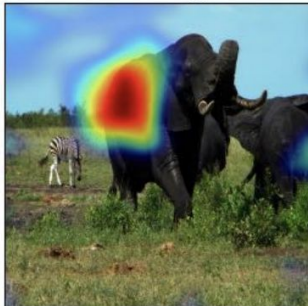
What is she holding?



Baseball bat



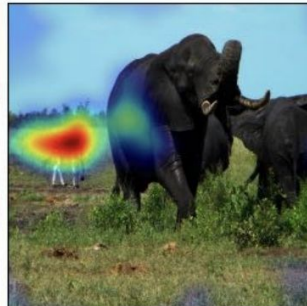
What is that?



Elephant



What is that?



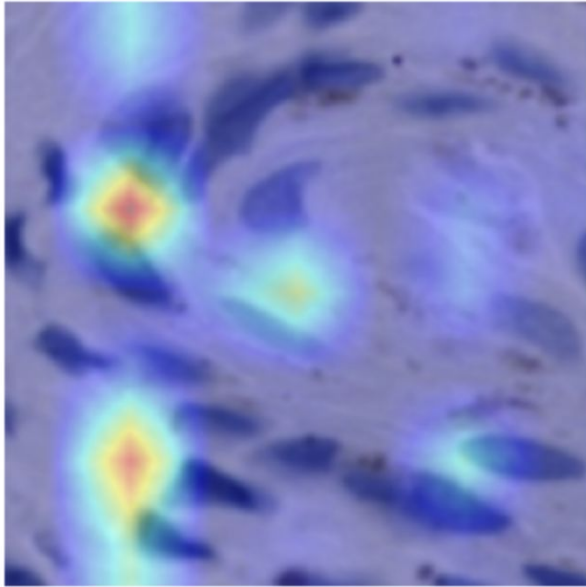
Zebra



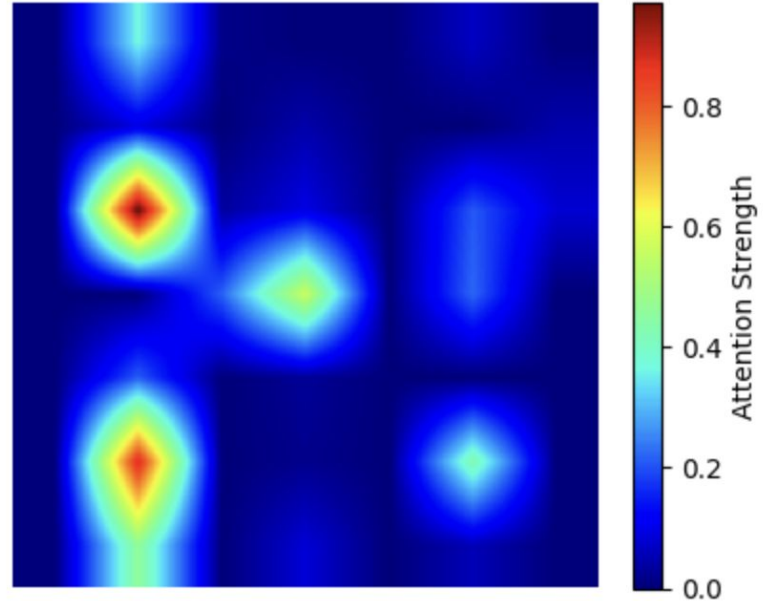
# Applications: Grad-CAM

Patch 1 - Benign

Grad-CAM Overlay



Attention Map (Color Scale)





# Next steps

- End-to-end toy example of attention
- Read and summarize paper on BAM
- Annotated snippet explaining spatial vs channel attention

# Sources

Vaswani, Ashish, et al. "Attention is All You Need." Advances in Neural Information Processing Systems, vol. 30, 2017. <https://arxiv.org/abs/1706.03762>

Xu, Kelvin, et al. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention." International Conference on Machine Learning, PMLR, 2015, pp. 2048–2057.  
<https://arxiv.org/abs/1502.03044>

Selvaraju, Ramprasaath R., et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization." Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 618–626. <https://arxiv.org/abs/1610.02391>

Woo, Sanghyun, et al. "CBAM: Convolutional Block Attention Module." Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19. <https://arxiv.org/abs/1807.06521>

<https://medium.com/biased-algorithms/attention-mechanism-for-image-classification-040a416a630d>

<https://www.analyticsvidhya.com/blog/2024/01/different-types-of-attention-mechanisms/>