# LambdaNetworks: Modeling long-range Interactions without Attention

Anonymous authors

Paper under double-blind review

# Outline

- Introduction
- Methodology
- Experiments
- Conclusion

# Introduction

Using Self Attention to obtain contextual information is indeed helpful to improve the accuracy of the model.

However, the amount of memory to be consumed makes it difficult to apply to very long sequences and multi-dimensional (such as images) tasks.

# Introduction

Therefore, the author proposes the architecture of lambda network, which can obtain contextual information while reducing memory consumption and increasing computing speed.

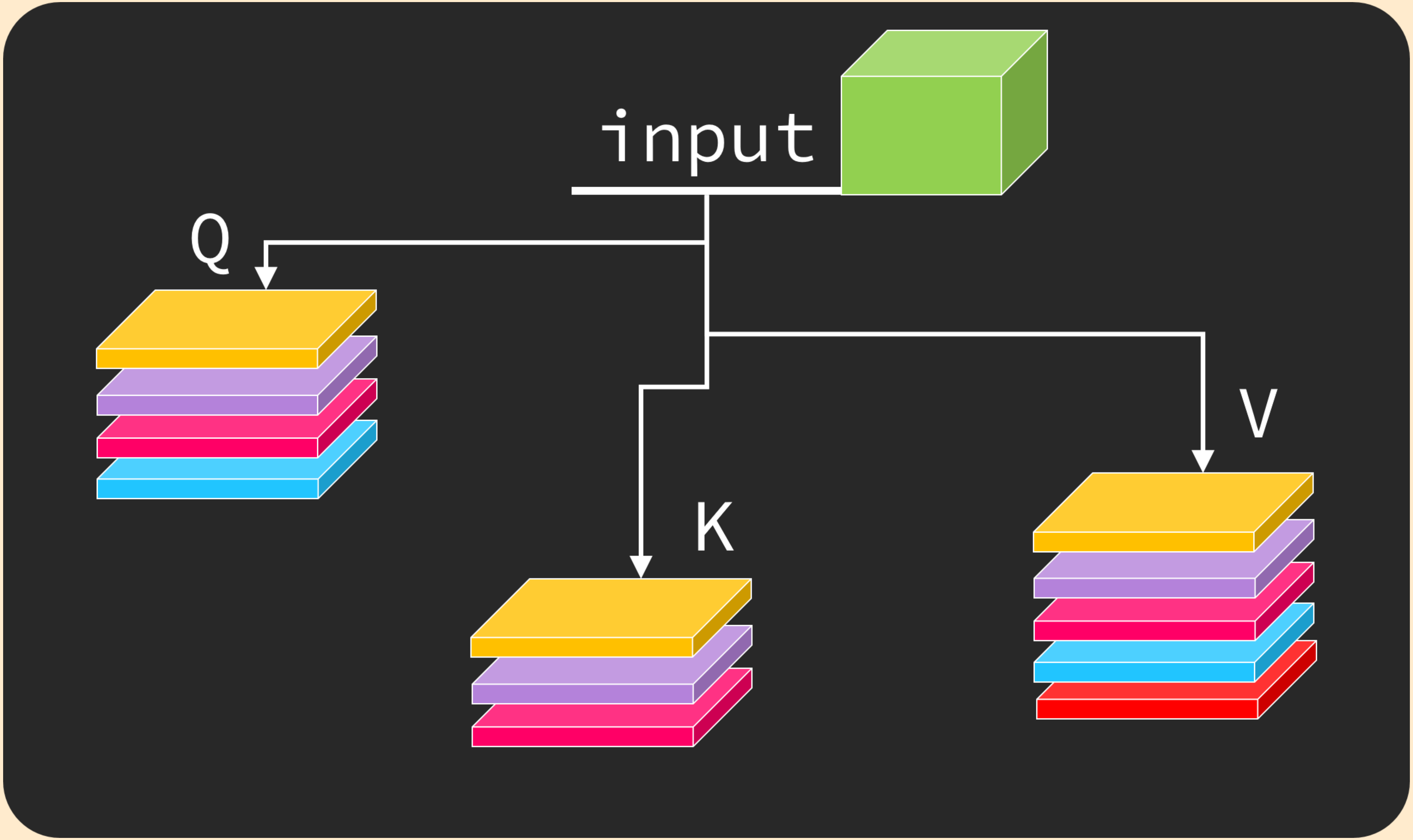
# Methodology

Content Lambda

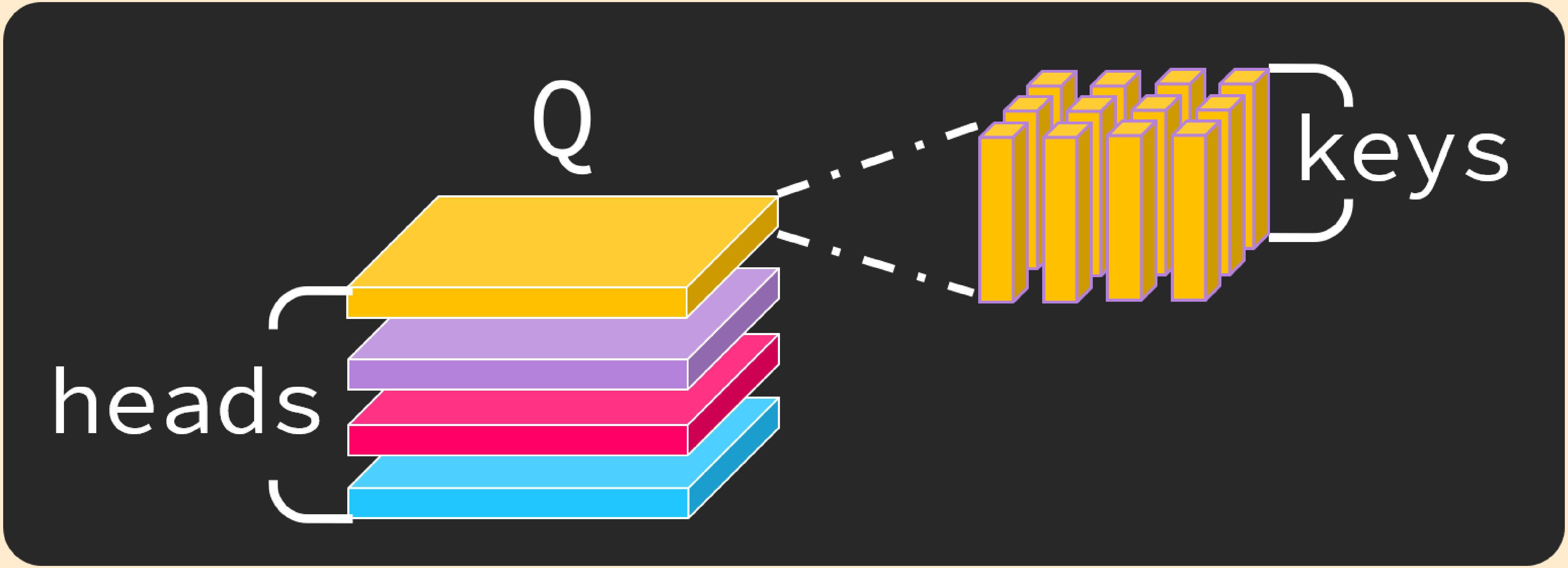
+

Position Lambda

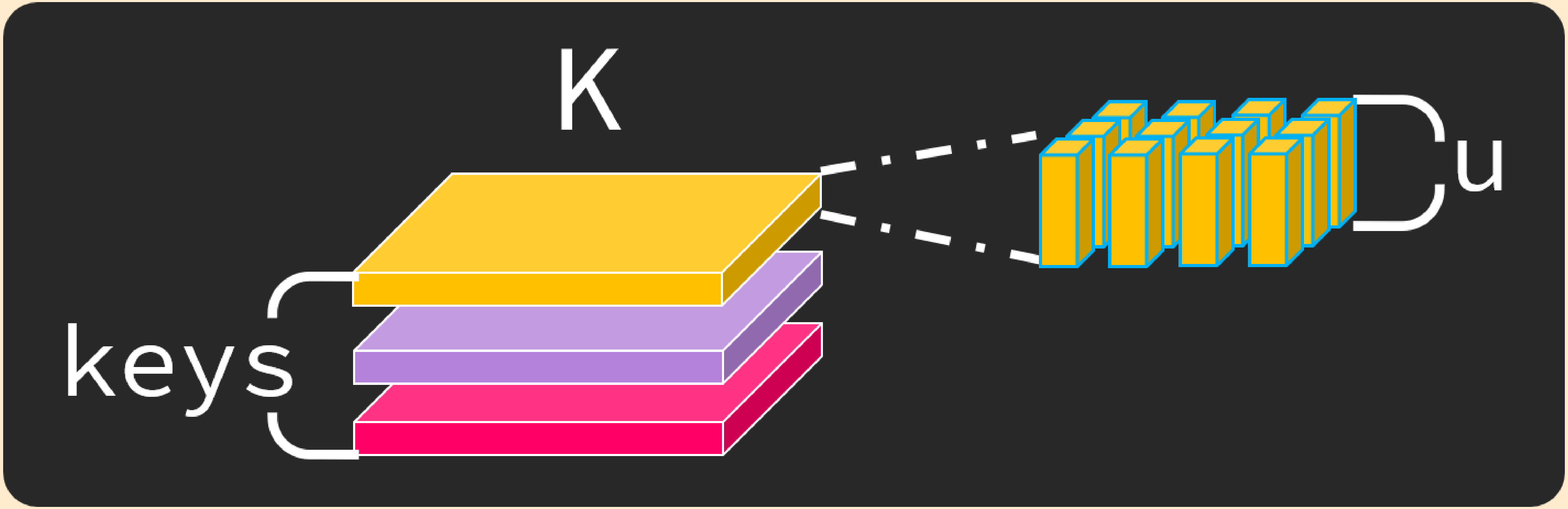
# Methodology



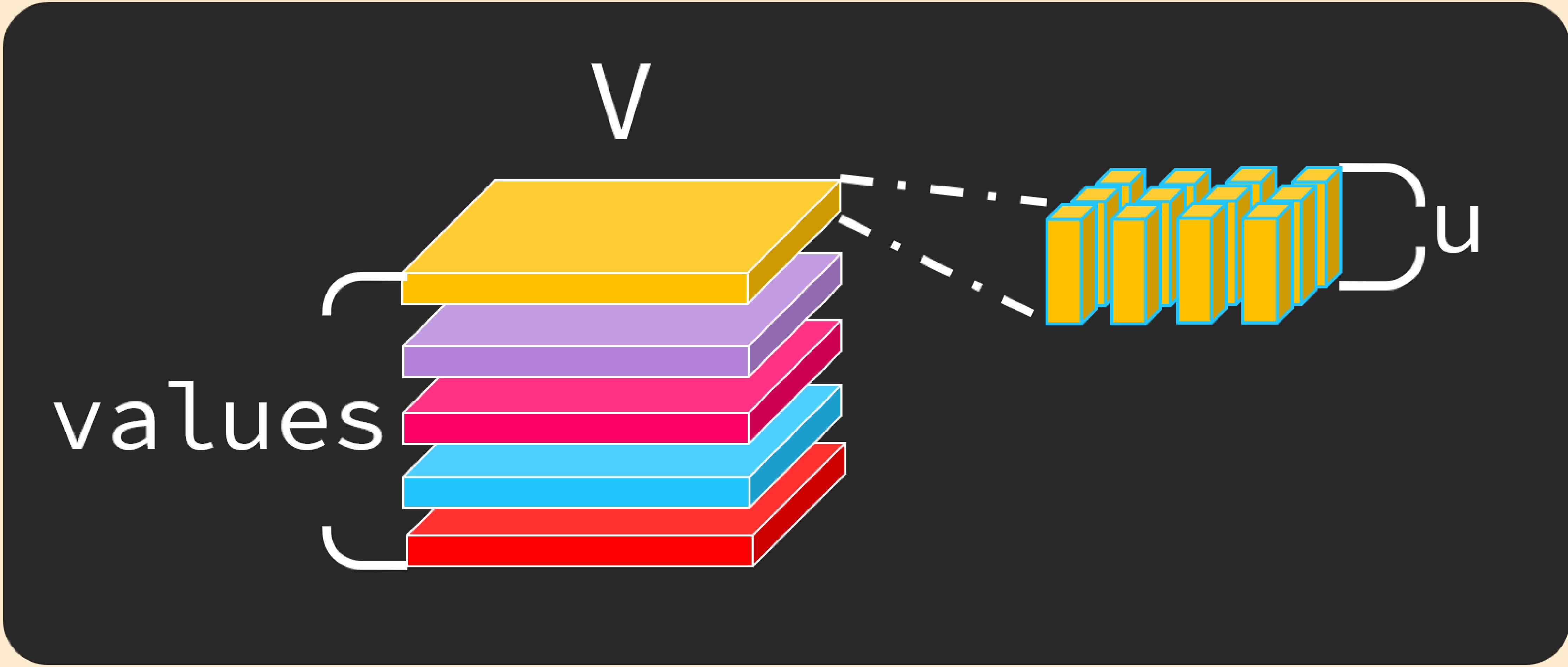
# Methodology



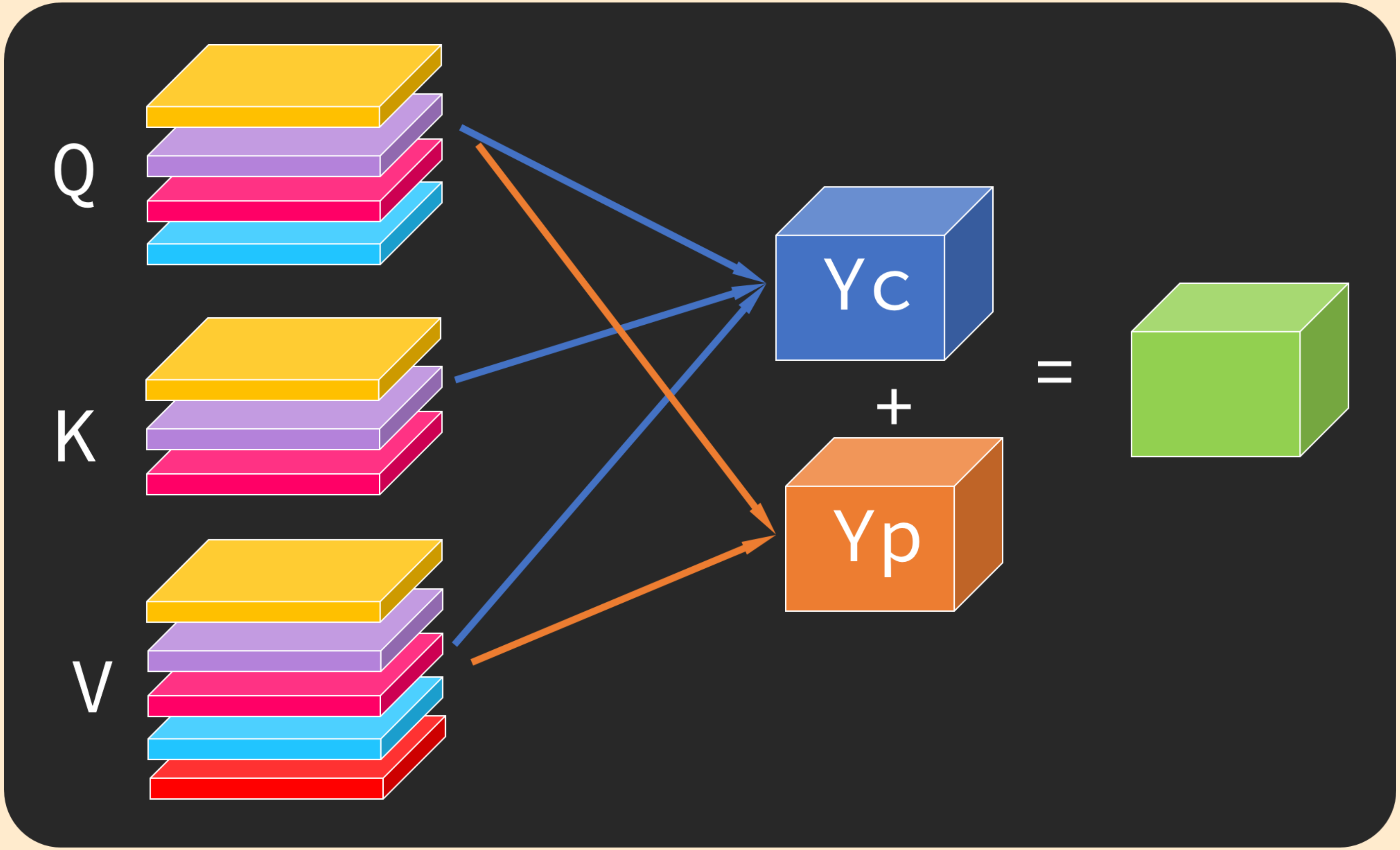
# Methodology



# Methodology



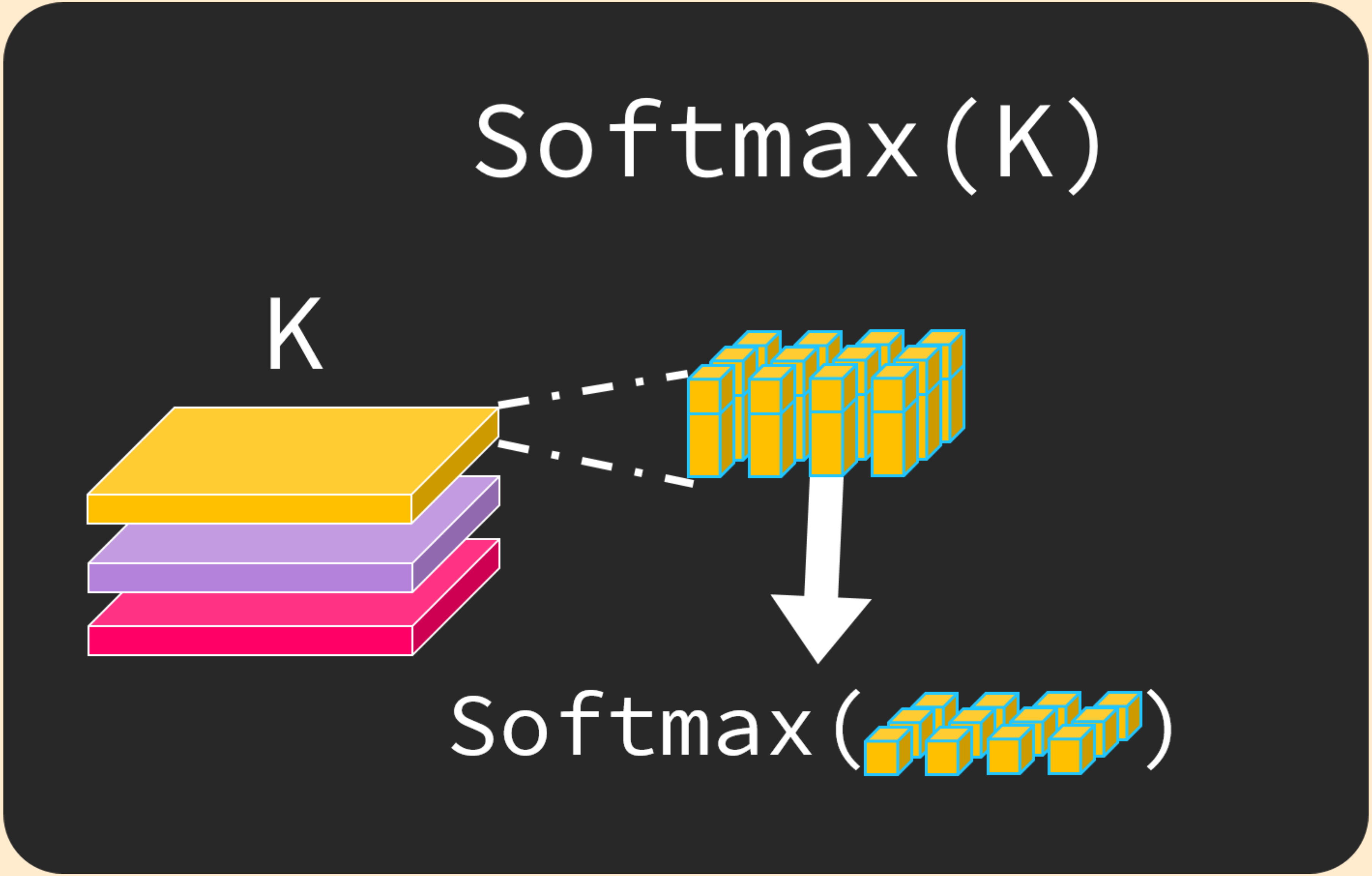
# Methodology



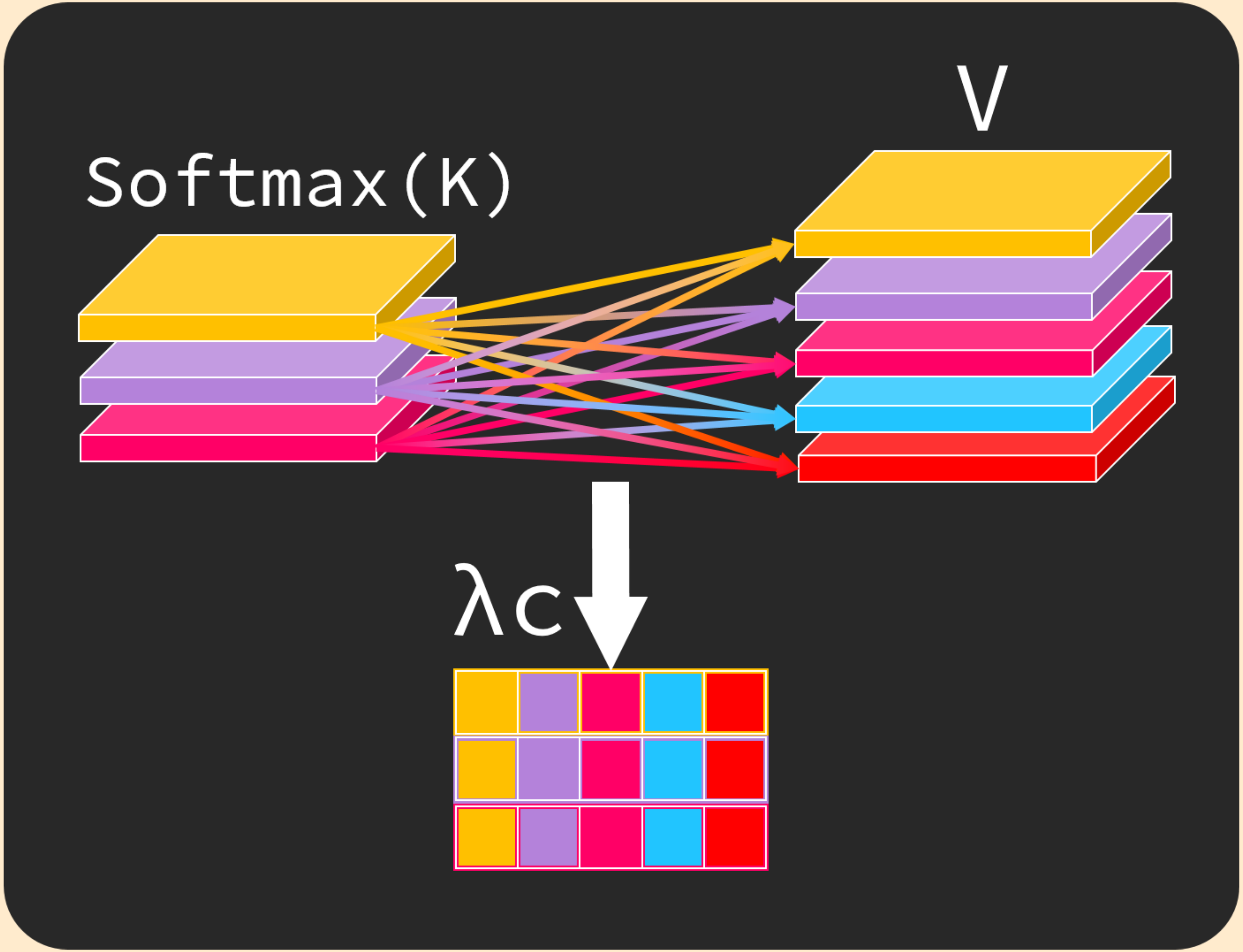
# Content Lambda

Matching to Feature「Map」

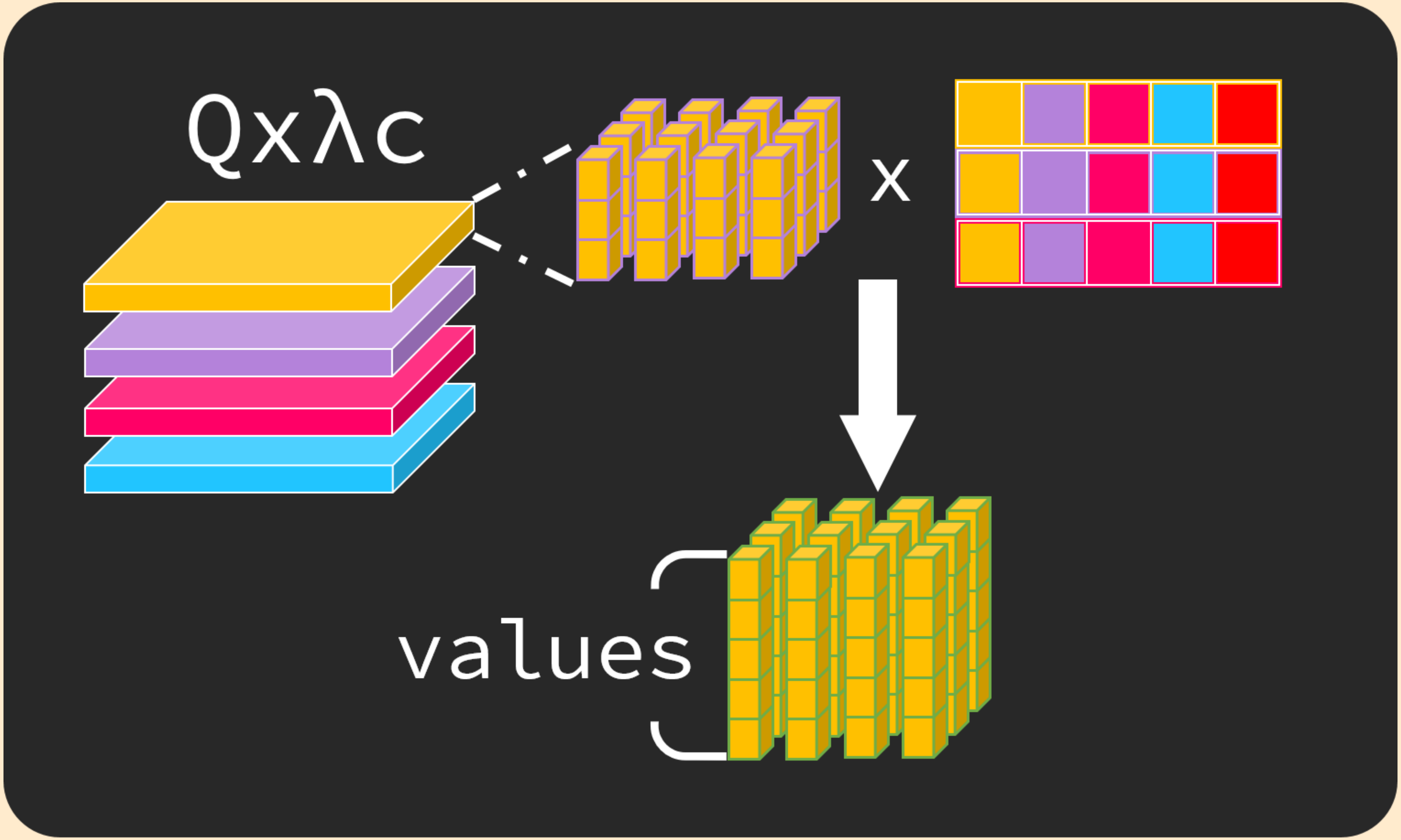
# Content Lambda



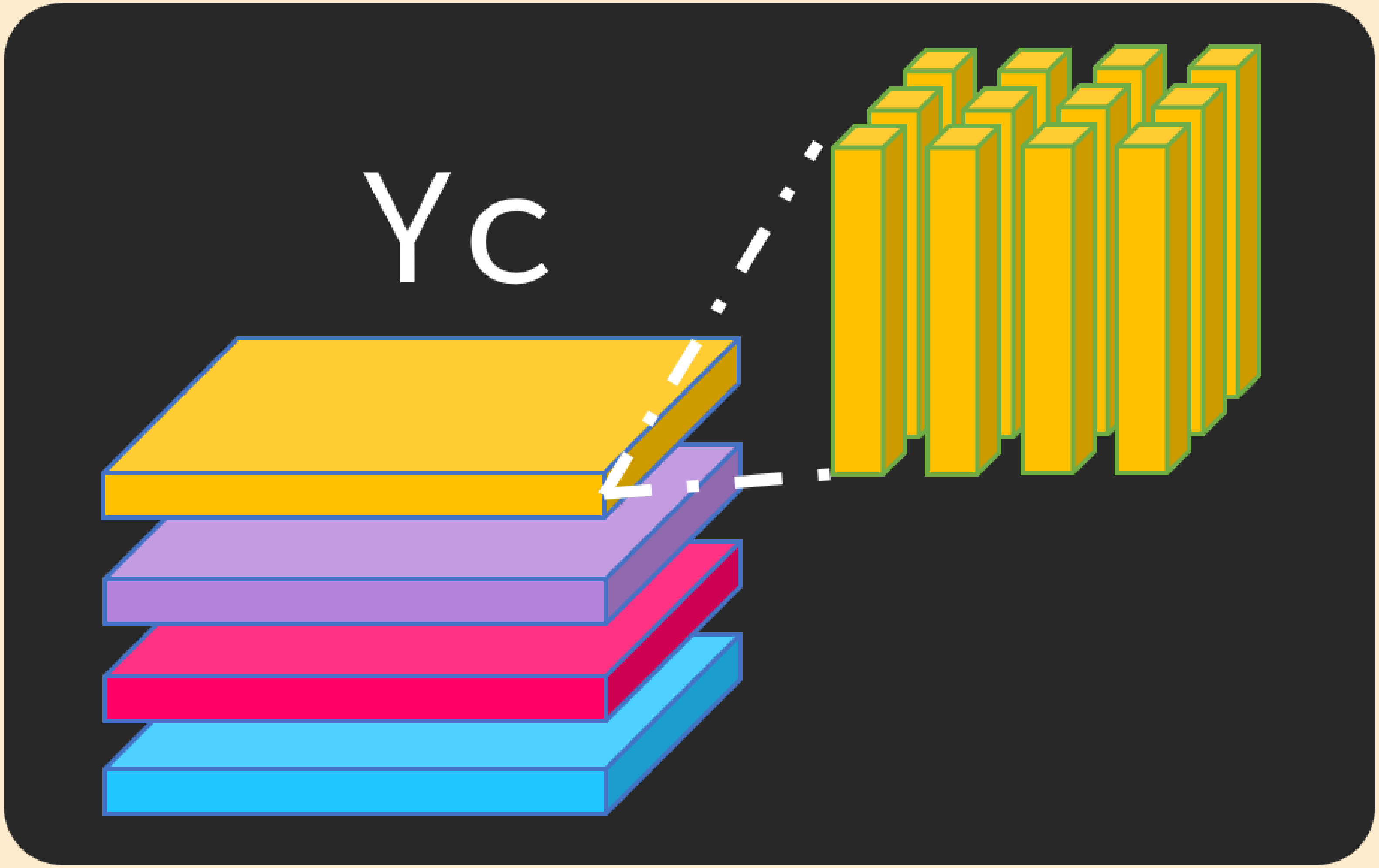
# Content Lambda



# Content Lambda



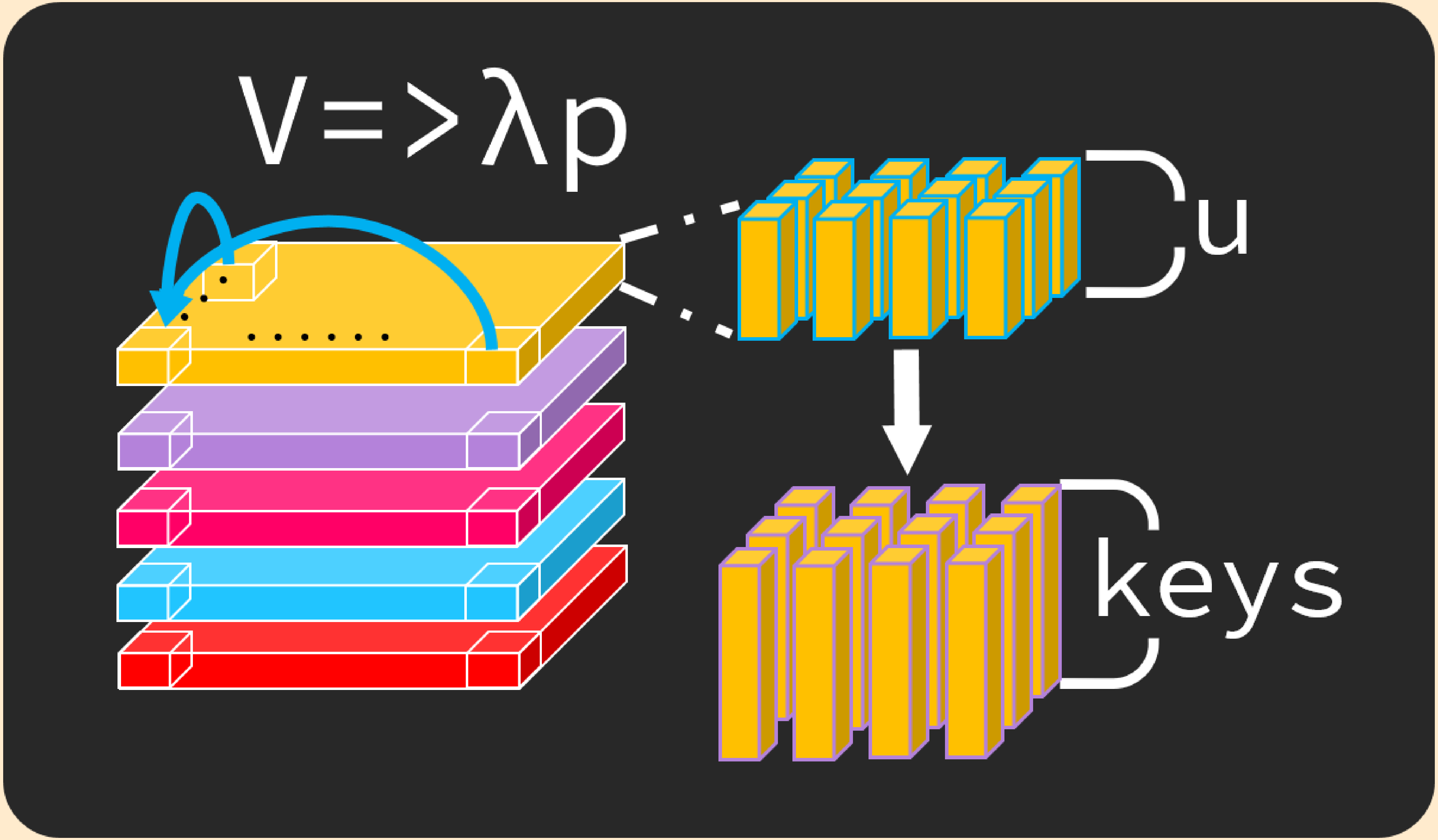
# Content Lambda



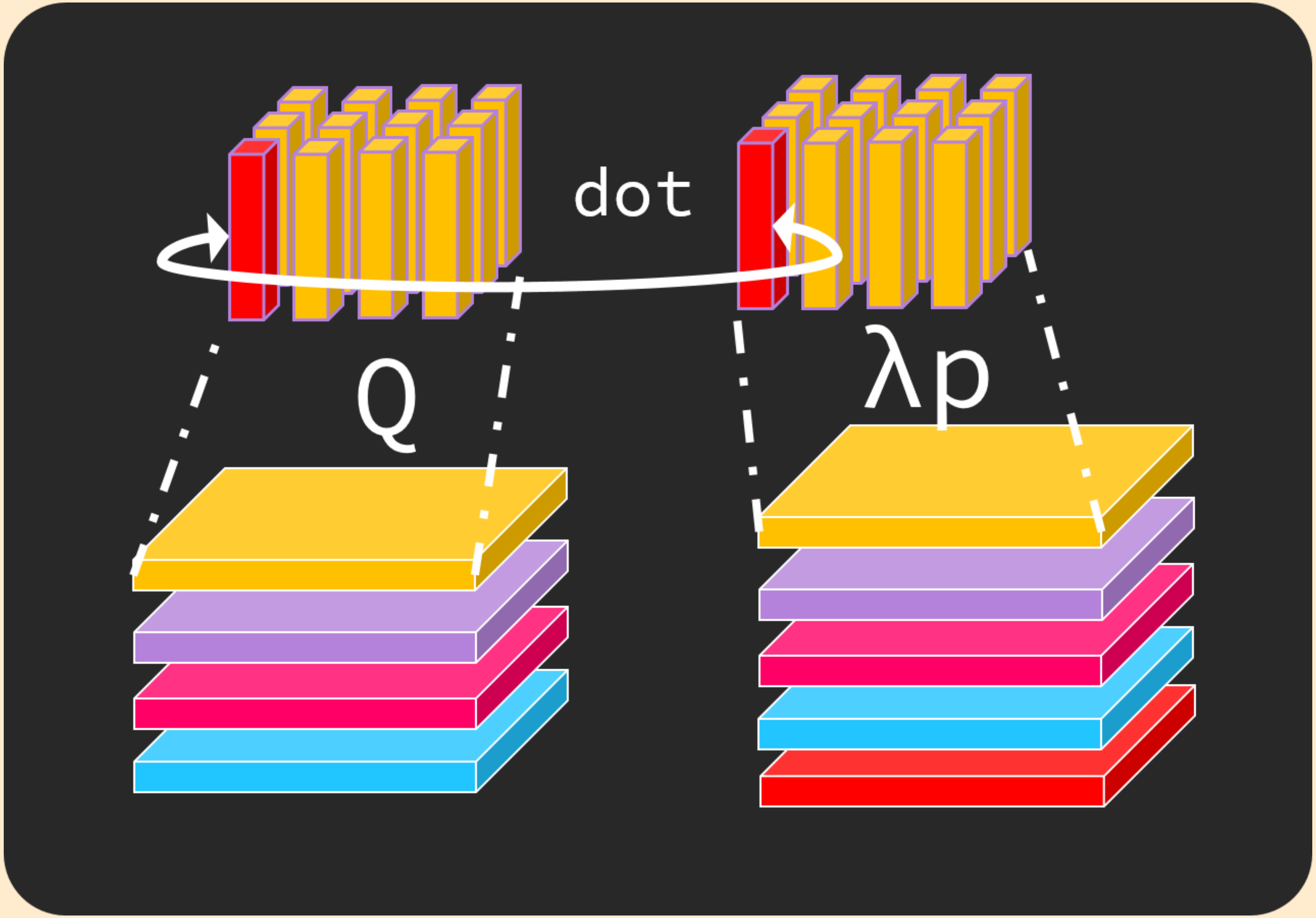
# Position Lambda

Matching to Feature「Vector」

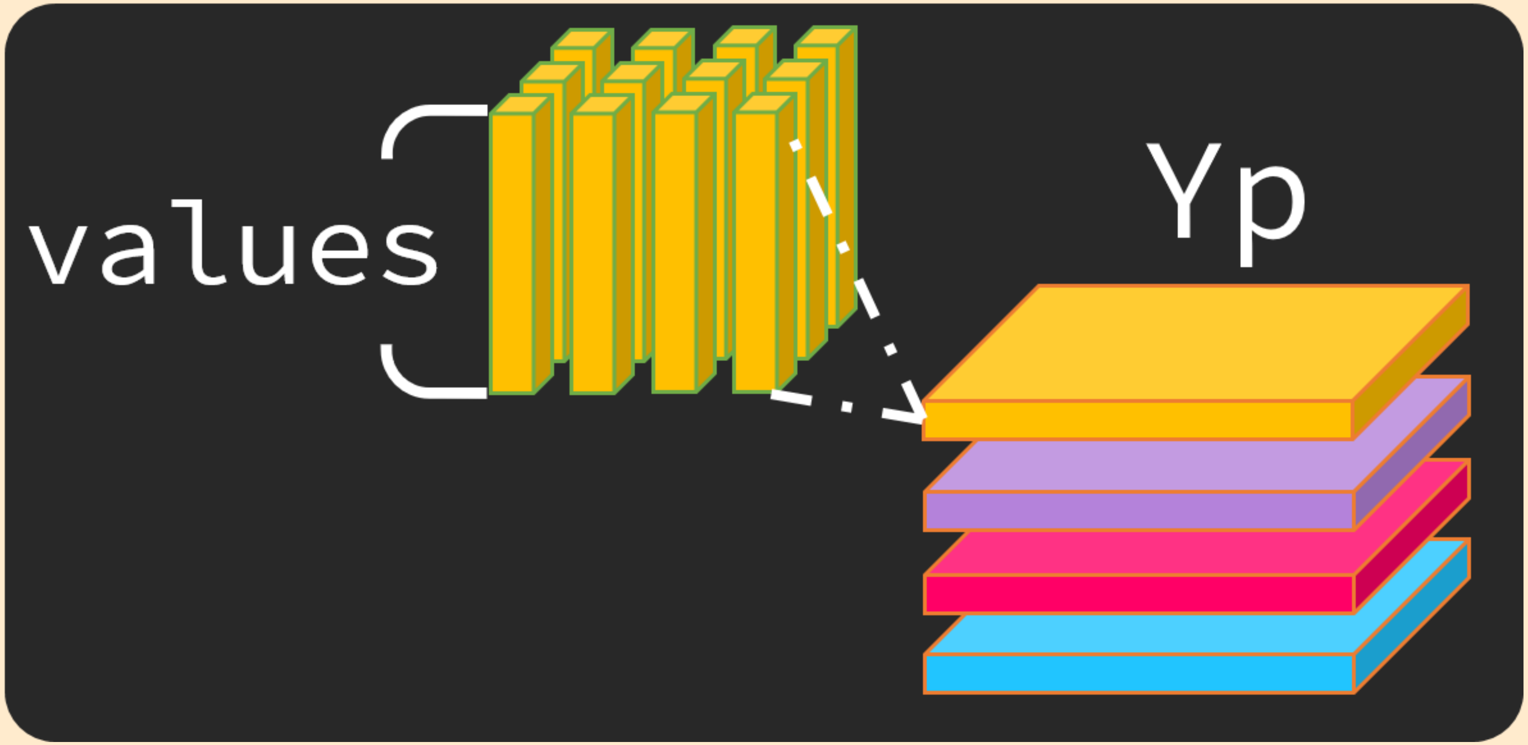
# Position Lambda



# Position Lambda



# Position Lambda



# Experiments

- vs Baseline
- Content vs Position
- Normalization
- Other

# Classification vs Baseline

| Layer | Params (M) | top-1 |
|---|---|---|
| Conv (He et al., 2016)† | 25.6 | $76.9_{+0.0}$ |
| Conv + channel attention (Hu et al., 2018b)† | 28.1 | $77.6_{+0.7}$ |
| Conv + linear attention (Chen et al., 2018) | 33.0 | 77.0 |
| Conv + linear attention (Shen et al., 2018) | - | $77.3_{+1.2}$ |
| Conv + relative self-attention (Bello et al., 2019) | 25.8 | $77.7_{+1.3}$ |
| Local relative self-attention (Ramachandran et al., 2019) | 18.0 | $77.4_{+0.5}$ |
| Local relative self-attention (Hu et al., 2019) | 23.3 | $77.3_{+1.0}$ |
| Local relative self-attention (Zhao et al., 2020) | 20.5 | $78.2_{+1.3}$ |
| Lambda layer | **15.0** | $\mathbf{78.4}_{+1.5}$ |
| Lambda layer ($\lvert u \rvert$=4) | **16.0** | $\mathbf{78.9}_{+2.0}$ |

# Detection



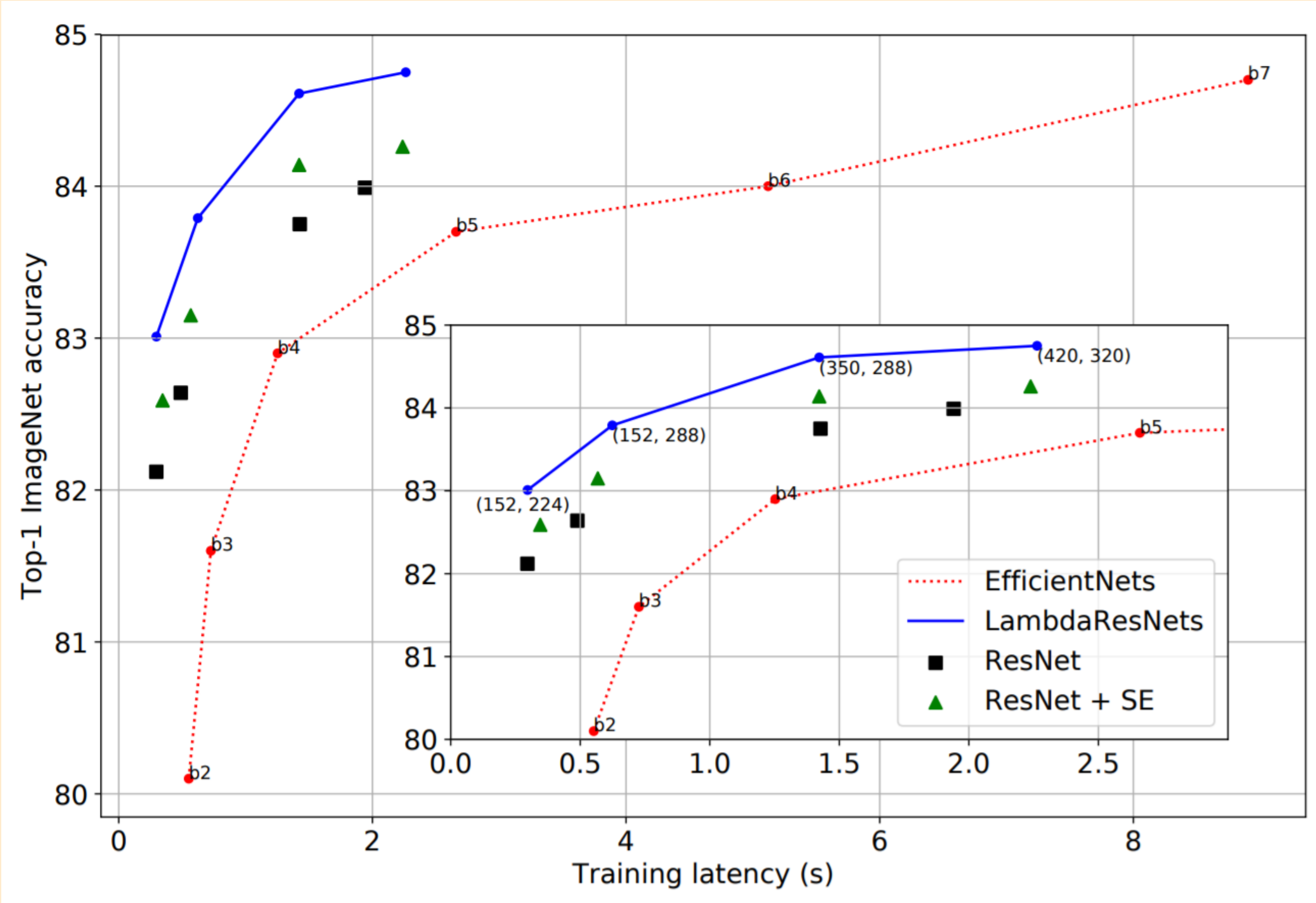
| Backbone | $AP^{bb}_{coco}$ | $AP^{bb}_{s/m/l}$ |
|---|---|---|
| ResNet-101 | 48.2 | 29.9 / 50.9 / 64.9 |
| ResNet-101 + SE | 48.5 | 29.9 / 51.5 / 65.3 |
| LambdaResNet-101 | **49.4** | **31.7 / 52.2 / 65.6** |
| ResNet-152 | 48.9 | 29.9 / 51.8 / 66.0 |
| ResNet-152 + SE | 49.4 | 30.0 / 52.3 / 66.7 |
| LambdaResNet-152 | **50.0** | **31.8 / 53.4 / 67.0** |

# Segmentation

vs Baseline

| Backbone | $AP^{mask}_{coco}$ | $AP^{mask}_{s/m/l}$ |
|---|---|---|
| ResNet-101 | 42.6 | 24.2 / 45.6 / 60.0 |
| ResNet-101 + SE | 42.8 | 24.0 / 46.0 / 60.2 |
| LambdaResNet-101 | **43.5** | **25.9 / 46.5 / 60.8** |
| ResNet-152 | 43.2 | 24.2 / 46.1 / 61.2 |
| ResNet-152 + SE | 43.5 | 24.6 / 46.8 / 61.8 |
| LambdaResNet-152 | **43.9** | **25.5 / 47.3 / 62.0** |

# Training vs Baseline



# Content vs Position

| Content | Position | Params (M) | FLOPS (B) | top-1 |
|---|---|---|---|---|
| ✓ | × | 14.9 | 5.0 | 68.8 |
| × | ✓ | 14.9 | 11.9 | 78.1 |
| ✓ | ✓ | 14.9 | 12.0 | 78.4 |

Position Lambda is more important than Content Lambda.

# Normalization

| Normalization | top-1 |
|---|---|
| Softmax on keys (default) | 78.4 |
| Softmax on keys and queries | 78.1 |
| L2-normalized keys | 78.0 |
| Non-normalized keys | 70.0 |
| No batch normalization on queries and values | 76.2 |

It is necessary to regulate K.

# Other

| Architecture | Params (M) | Throughput | top-1 |
|---|---|---|---|
| $\mathbf{C} \to \mathbf{C} \to \mathbf{C} \to \mathbf{C}$ | 25.6 | 7240ex/s | 76.9 |
| $\mathbf{L} \to \mathbf{C} \to \mathbf{C} \to \mathbf{C}$ | 25.5 | 1880ex/s | 77.3 |
| $\mathbf{L} \to \mathbf{L} \to \mathbf{C} \to \mathbf{C}$ | 25.0 | 1280ex/s | 77.2 |
| $\mathbf{L} \to \mathbf{L} \to \mathbf{L} \to \mathbf{C}$ | 21.7 | 1160ex/s | 77.8 |
| $\mathbf{L} \to \mathbf{L} \to \mathbf{L} \to \mathbf{L}$ | 15.0 | 1160ex/s | 78.4 |
| $\mathbf{C} \to \mathbf{L} \to \mathbf{L} \to \mathbf{L}$ | 15.1 | 2200ex/s | 78.3 |
| $\mathbf{C} \to \mathbf{C} \to \mathbf{L} \to \mathbf{L}$ | 15.4 | 4980ex/s | 78.3 |
| $\mathbf{C} \to \mathbf{C} \to \mathbf{C} \to \mathbf{L}$ | 18.8 | 7160ex/s | 77.3 |

Lambda Layer will have better results after Convolution.

# Other

| Layer | Complexity | Memory (GB) | Throughput | top-1 |
|---|---|---|---|---|
| Global self-attention | $\Theta(blhn^2)$ | 120 | OOM | OOM |
| Axial self-attention | $\Theta(blhn\sqrt{n})$ | 4.8 | 960ex/s | 77.5 |
| Local self-attention (7x7) | $\Theta(blhnm)$ | - | 440ex/s | 77.4 |
| Lambda layer | $\Theta(lkn^2)$ | 0.96 | 1160ex/s | **78.4** |
| Lambda layer ($\|k\|$=8) | $\Theta(lkn^2)$ | 0.48 | **1640**ex/s | 77.9 |
| Lambda layer (shared embeddings) | $\Theta(kn^2)$ | 0.31 | 1210ex/s | 78.0 |
| Lambda convolution (7x7) | $\Theta(lknm)$ | - | 1100ex/s | 78.1 |

Lambda has higher speed, accuracy and lower memory consumption than Self Attention.

# Other

| Config | Params (M) | Throughput | top-1 |
|---|---|---|---|
| ResNet101 - 224x224 | | | |
| Baseline | 44.6 | 4600 ex/s | 81.3 |
| + SE | 63.6 | 4000 ex/s | 81.8 |
| + 3 lambda | 36.9 | 4040 ex/s | 82.3 |
| + all lambdas | 26.0 | 2560 ex/s | 82.6 |
| ResNet152 - 256x256 | | | |
| Baseline | 60.2 | 2780 ex/s | 82.5 |
| + SE | 86.6 | 2400 ex/s | 83.0 |
| + 6 lambdas | 51.4 | 2400 ex/s | 83.4 |
| + all lambdas | 35.1 | 1480 ex/s | 83.4 |

# Receptive Field



| Scope size $\|m\|$ | 3x3 | 7x7 | 15x15 | 23x23 | 31x31 | global |
|---|---|---|---|---|---|---|
| FLOPS (B) | 5.7 | 6.1 | 7.8 | 10.0 | 12.4 | 19.4 |
| Top-1 Accuracy | 77.6 | 78.2 | 78.5 | 78.3 | 78.5 | 78.4 |

In the experiment, the receptive field of Position Lambda is not the bigger the better.

# Conclusion

- Lambda Layer can be understood as a better Channel + Spatial Attention.
- Compared with Linear Attention, Lambda Layer has the ability to focus better position.
- Lighter and faster than Self Attention.