Authors propose **transpose of attention**

- **Solution - transposed** self-attention that operates **across feature channels**, not tokens
    - **Effectively a dynamic 1x1 convolution**
- We **no longer have quadratic complexity** in the length of input sequence
    - This means it's really good for **Image Transformers** that work on patches
- It's more or less a **convolutional neural network with one dynamic layer to it**
    - One of the convolutions is a **dynamic convolution**
- *Self-attention yields global interactions between all tokens, e.g. words or image patches*
    - Allows for **flexible modelling of image data beyond local interactions** (i.e. CNNs)
        - **Problem -** flexibility results in **quadratic complexity** in time and memory, bad for long sequences
- **Solution - transposed** self-attention that operates **across feature channels**, not tokens
- **Cross-covariance attention XCA -** has linear complexity in number of tokens
- **Coss-Covariance Image Transformer XCIT -** accuracy close to transformers with more scalability

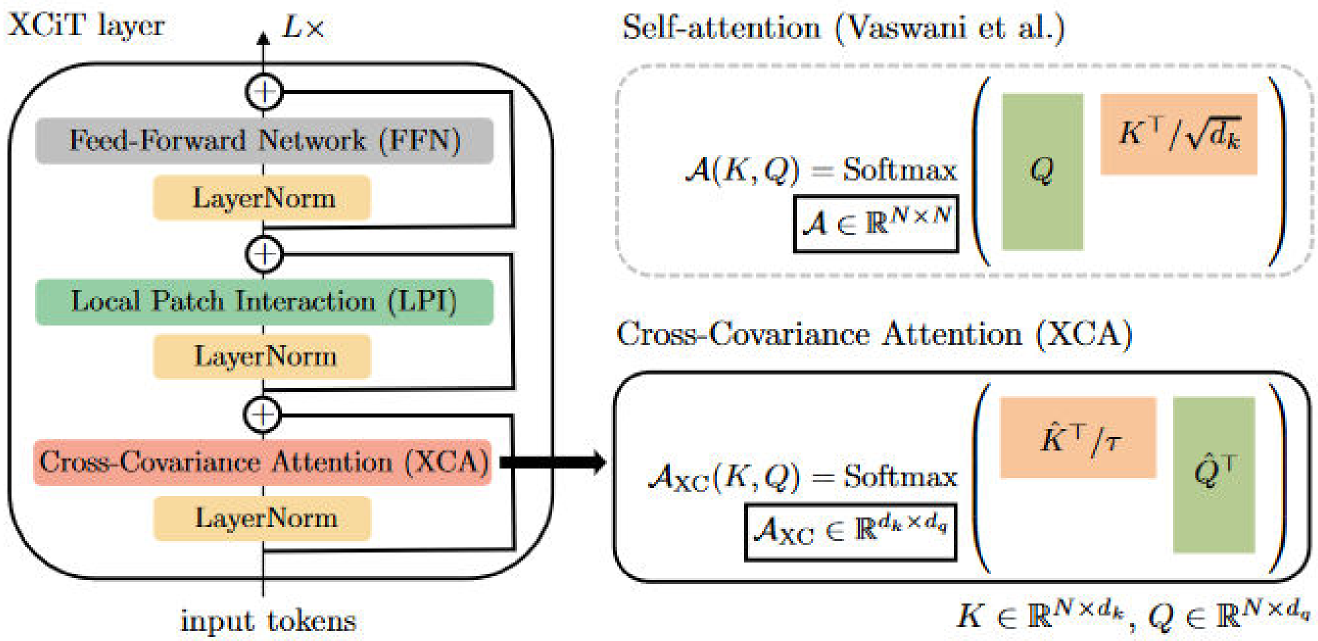
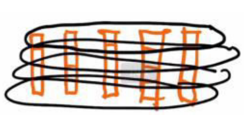


Figure 1: Our XCiT layer consists of three main blocks, each preceded by LayerNorm and followed by a residual connection: (i) the core cross-covariance attention (XCA) operation, (ii) the local patch interaction (LPI) module, and (iii) a feed-forward network (FFN). By transposing the query-key interaction, the computational complexity of XCA is linear in the number of data elements N, rather than quadratic as in conventional self-attention.

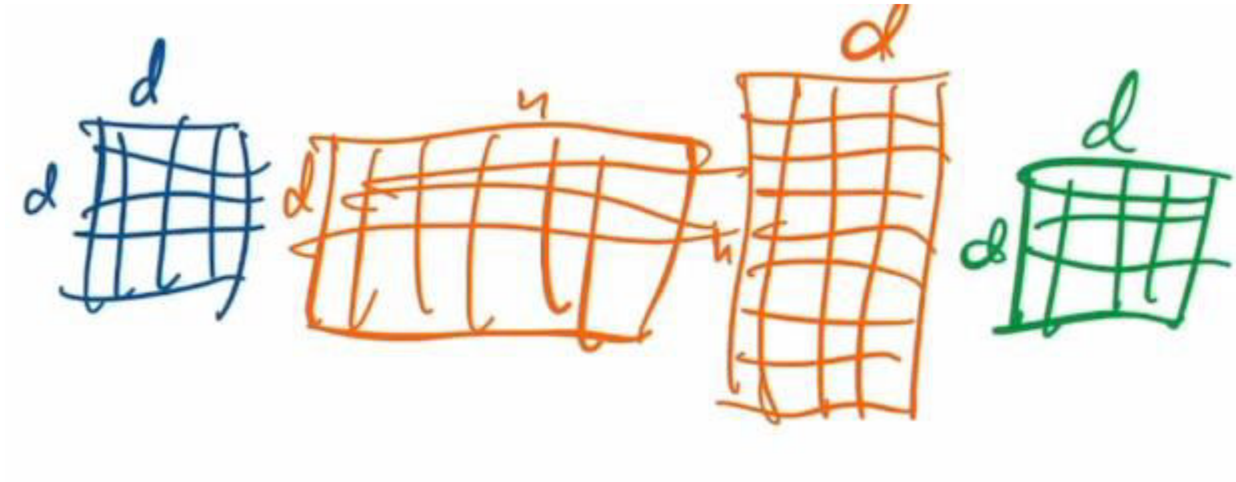
- The whole model consists of **XCiT layers**
- You have **L** XCiT blocks
- You then have a classification layer / segmentation layer on top
- *The self-attention has been replaced by a **Cross-Covariance XCA** block and a **Local Patch Interaction** block*

Basic Idea 

- Instead of having a sequence of vectors, you transpose the input so that each vector of a single channel across all inputs is an input to the network instead instead
- **Each channel exposes a query and a key**
- You look across the entire sequence in each channel
- *Instead of determining what is in a single patch, the new inputs communicate what feature is where*
    - Instead of asking what is a patch, you ask **where is the mouth**

Cross Attention - You multiply the matrices in the opposite order

- So instead of $\mathbf{X.W_Q W_K{}^T.X^T}$, you are doing the cross product $\mathbf{W_Q.X.X^T.W_K}$ which happens to have a smaller dim
- *These matrices are now multiplied in a different order*
- The resulting matrix is **d x d** instead of **n x n**



Local Patch Interaction LPI - It's a **convolution**

- In our sequence of tokens, you go through the **XCiT** layer, then take the output and **slide a kernel over it**
- It's **depth separated**, you have a **1D kernel** (very few parameters, so little memory overhead)

Considerations (things you have to do)

- Do **L2 - Normalization -** It breaks down without this
- Do **Temperature Scaling -** Learned temperature parameter
- Do **Block-Diagonal Cross-Covariance Attention -** They don't attend from all channels to all channels, they have this block where a channel can only attend to nearby channels

You don't really have attention between patches so it doesn't really behave like a normal transformer… Is a transformer anything with dynamic weights?

- No long range information exchange
- More of an evolution of CNNs