**Residual Connections and Normalization in Neural Networks (Focus: Transformers)**

**1. Residual Connections**

A **residual connection**, also known as a **skip connection**, is a key architectural feature in deep networks like Transformers and ResNets. The idea is to add the original input to the output of a layer:

Y=F(X)+XY = F(X) + X

Where:

* **X**: input to the layer
* **F(X)**: transformation applied by the layer (e.g., self-attention, feed-forward)
* **Y**: result after adding the input (residual)

**Importance:**

* **Mitigates the vanishing gradient problem**: Direct gradient flow through shortcut paths.
* **Enables deeper architectures**: Easier optimization.
* **Improves convergence**: Model trains faster and more reliably.

**Where used in Transformers:**

* Across **self-attention layers**
* Across **feed-forward layers**

Each of these is followed by a residual addition and normalization step: **Add & Norm**.

**2. Add & Norm Layer**

This consists of two operations:

* **Add**: Residual connection adds the input to the layer's output.
* **Norm**: Layer normalization is applied to the resulting vector.

**3. Layer Normalization**

Normalization is crucial to stabilize training in deep networks. **Layer Normalization** normalizes the elements **within a single vector**:

x^i=xi−μσ\hat{x}\_i = \frac{x\_i - \mu}{\sigma}

Where:

* μ\mu: Mean of the vector's elements
* σ\sigma: Standard deviation of the vector's elements

**Why Layer Normalization?**

* Applied **independently to each instance**, unlike Batch Norm which works across a batch.
* Better suited for sequence modeling and NLP tasks where batch statistics may vary.

**4. Why Normalization is Needed**

During deep transformations:

* Feature distributions change, leading to unstable gradients
* Output ranges vary unpredictably

**Benefits of Normalization:**

* **Stabilizes and speeds up training**
* **Reduces internal covariate shift**
* **Prevents exploding/vanishing gradients**
* **Improves generalization (less overfitting)**

**5. Normalization vs Standardization**

| **Term** | **Description** | **Output Characteristics** |
| --- | --- | --- |
| Normalization | Rescale inputs to a fixed range (e.g., 0 to 1) | Values lie in a specific bounded interval |
| Standardization | Rescale inputs to have mean = 0, std = 1 | Values follow a standard normal distribution |

Often used interchangeably, but they differ slightly in goal and method.

**6. Example: Why Normalize?**

Given two features:

* **Age**: ranges from 18 to 25
* **Tuition Fees**: ranges from $20k to $50k

If unnormalized, the large scale of tuition fees can dominate the training dynamics. Solution:

* **Min-Max Normalization**: x−xminxmax−xmin\frac{x - x\_{min}}{x\_{max} - x\_{min}}
* **Standardization**: x−μσ\frac{x - \mu}{\sigma}

**7. Batch Normalization**

**Batch Normalization (BN)** normalizes **across the batch**:

* A batch contains multiple instances
* For each **feature index** across instances, compute the mean and variance
* Normalize each feature across the batch

x′=x−μbatchσbatchx' = \frac{x - \mu\_{batch}}{\sigma\_{batch}}

**Limitations of BN in NLP:**

* Less effective when batch sizes are small
* Requires consistent batch statistics (not ideal for variable-length sequences)
* Layer Norm preferred for Transformers

**8. Comparison: Batch Norm vs Layer Norm**

| **Property** | **Batch Norm** | **Layer Norm** |
| --- | --- | --- |
| Normalization across | Batch (dimension-wise) | Features (per instance) |
| Best for | Vision, large batch sizes | NLP, small batches |
| Used in Transformers | Rarely | Frequently |

**9. Impact on Training**

* **Convergence Speed**: Faster with normalization
* **Gradient Flow**: More stable across layers
* **Overfitting**: Reduced, especially when combined with dropout

**10. Transformer Architecture Recap (Add & Norm Usage)**

In both encoder and decoder layers:

* After **multi-head self-attention** → Add & Norm
* After **feed-forward layers** → Add & Norm
* After **cross-attention** (decoder) → Add & Norm

These patterns make Transformers easier to train and more robust.

**Summary**

* **Residual connections** enable better gradient flow.
* **Add & Norm** combines residual addition and layer normalization.
* **Layer normalization** is ideal for Transformers, ensuring consistent training dynamics.
* **Normalization** is key for faster convergence, stable training, and better generalization.

End of Notes.