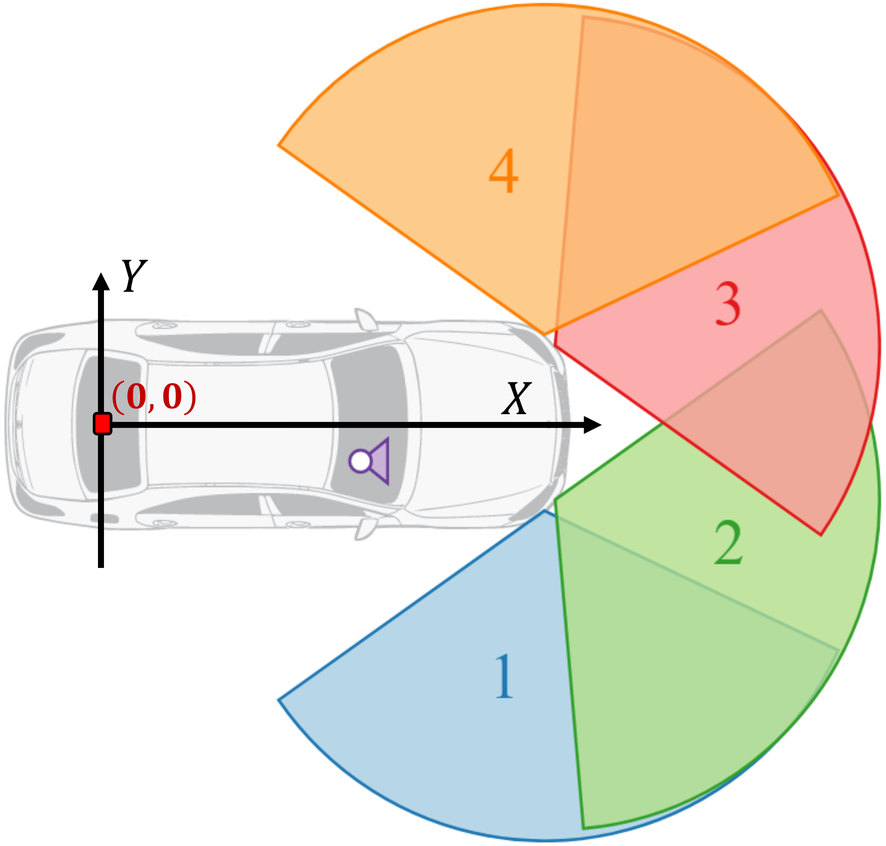
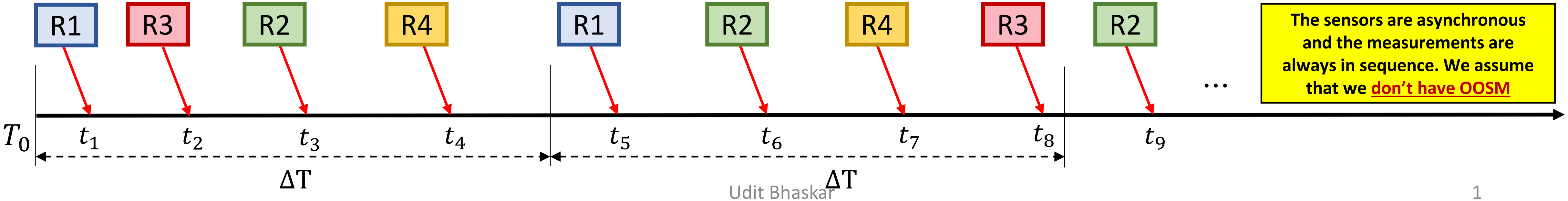
Sensor Setup

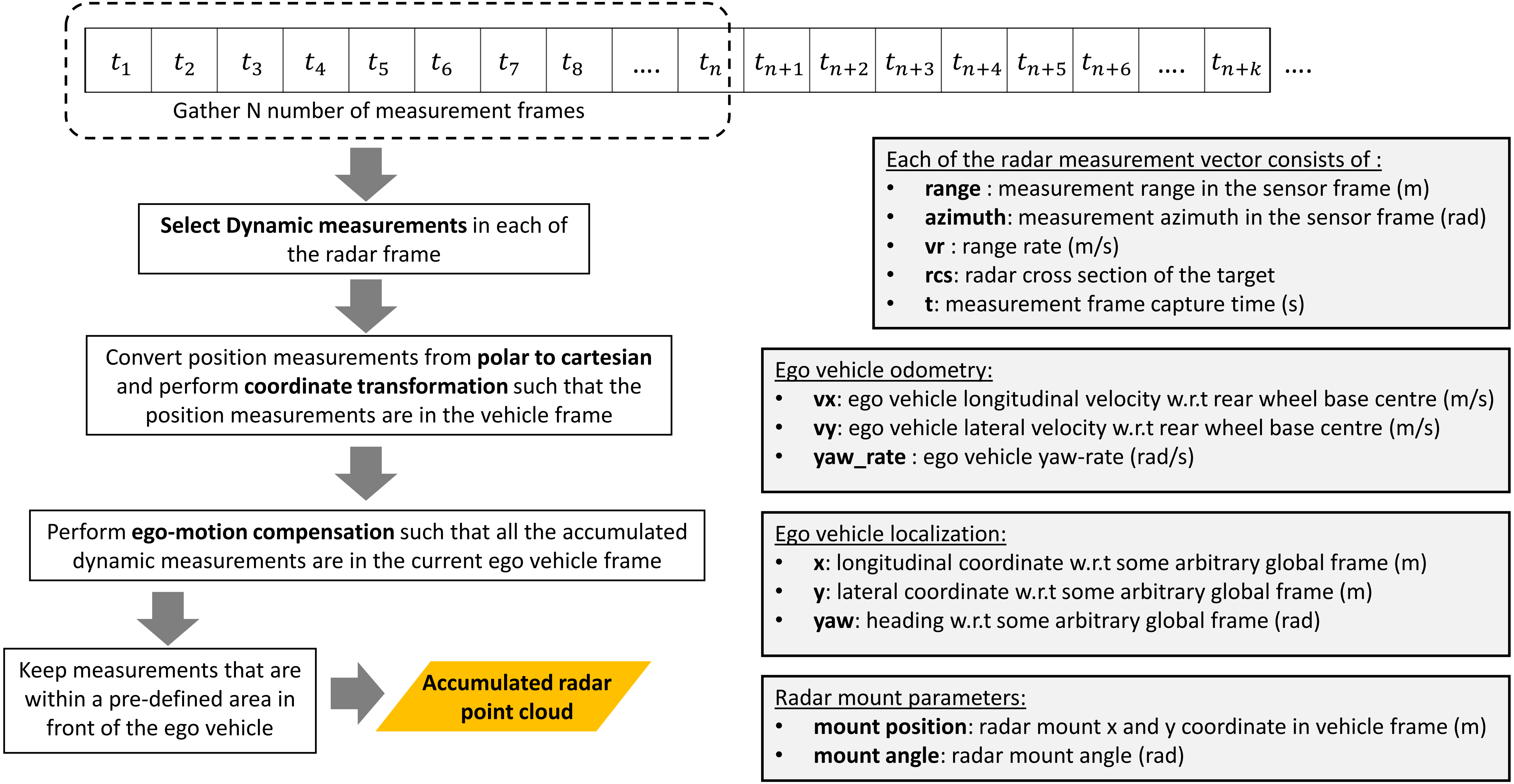
Source: *https://radar-scenes.com/dataset/sensors/*

Parameters / Sensor	Radar 1	Radar 2	Radar 3	Radar 4
Mount x coordinate	$+3.663$	$+3.86$	$+3.86$	$+3.663$
Mount y coordinate	-0.873	-0.7	$+0.7$	$+0.873$
Mount angle	$-85°$	$-25°$	$+25°$	$+85°$
Range resolution	0.15 meters			
Azimuth resolution	At the boresight direction, the resolution is about 0.5° and degrades to 2° at the outer parts of the field of view			
Range rate resolution	0.1 km/hr			
Maximum range	100 meters			
Maximum azimuth	$\pm 60°$			
Approximate measurement cycle	60 millisecond (approx. 17 Hz)			





Input Data Pre-processing



Dynamic Measurement Identification

input: Radar mount parameters

$\left(r_x^{mount}, r_y^{mount}, \theta^{mount}\right)$

input: ego motion

$\left(v_x^{ego}, v_y^{ego}, \dot{\omega}^{ego}\right)$

compute ego motion at the radar location in the radar frame

$$\mathrm{R} = \begin{bmatrix} \cos(\theta^{\mathrm{mount}}) & -\sin(\theta^{\mathrm{mount}}) \\ \sin(\theta^{\mathrm{mount}}) & \cos(\theta^{\mathrm{mount}}) \end{bmatrix}$$

$$\begin{bmatrix} \mathrm{v}_{\mathrm{x}}^{\mathrm{rad\_pred}} \\ \mathrm{v}_{\mathrm{y}}^{\mathrm{rad\_pred}} \end{bmatrix} = \mathrm{R}^{-1} \begin{bmatrix} 1 & 0 & -\mathrm{r}_{\mathrm{y}}^{\mathrm{mount}} \\ 0 & 1 & \mathrm{r}_{\mathrm{x}}^{\mathrm{mount}} \end{bmatrix} \begin{bmatrix} \mathrm{v}_{\mathrm{x}}^{\mathrm{ego\_pred}} \\ \mathrm{v}_{\mathrm{y}}^{\mathrm{ego\_pred}} \\ \dot{\omega}^{\mathrm{ego\_pred}} \end{bmatrix}$$

$\left(v_x^{rad\_pred}, v_y^{rad\_pred}\right)$

input: azimuth angles of the Radar measurements

θ^i

for each of the locations corresponding to radar measurements compute the predicted range rates

$$\mathrm{v}_{\mathrm{r\_pred}}^{\mathrm{i}} = -\left(\mathrm{v}_{\mathrm{x}}^{\mathrm{rad\_pred}} \cos\left(\theta^{\mathrm{i}}\right) + \mathrm{v}_{\mathrm{y}}^{\mathrm{rad\_pred}} \sin\left(\theta^{\mathrm{i}}\right)\right)$$

$v_{r\_pred}^i$

input: range rates of the Radar measurements

v_r^i

Compare the predicted range rate with the measurement range rate

$$\mathrm{error}^{\mathrm{i}} = \left|\mathrm{v}_{\mathrm{r\_pred}}^{\mathrm{i}} - \mathrm{v}_{\mathrm{r}}^{\mathrm{i}}\right|$$

$error^i$

Dynamic measurement selection

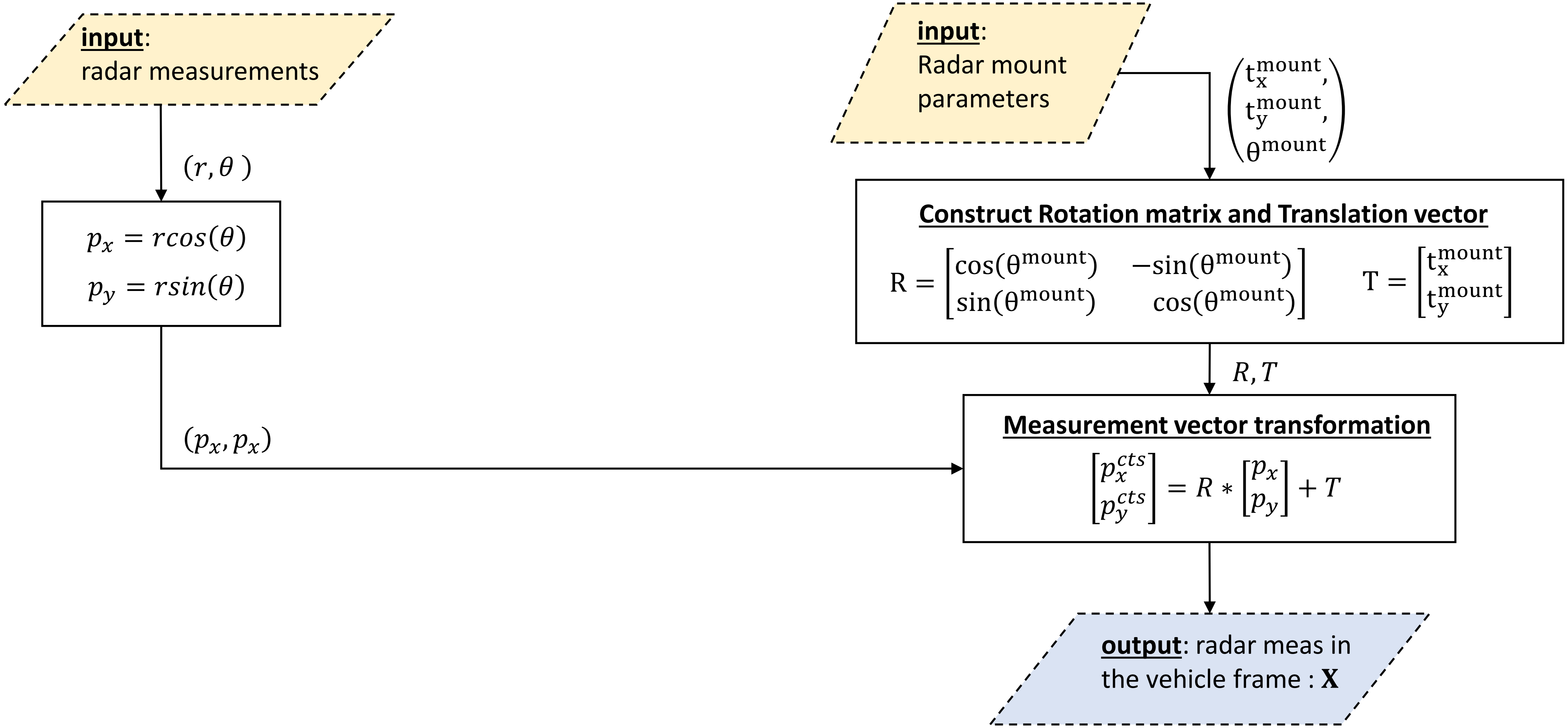
if $\mathrm{error}^{\mathrm{i}} \geq \mathit{threshold}$,
then Z^{i} is considered stationary

$Z_{dynamic}^i$

output: dynamic measurements

Polar to Cartesian measurement conversion

Coordinate Transformation Sensor to Vehicle Frame



Ego-motion Compensation

<u>input</u>:
- Ego pose at t-k
- Ego pose at t

$\left(px_t^{loc}, py_t^{loc}, \theta_t^{loc}\right)$ $\left(px_{t-k}^{loc}, py_{t-k}^{loc}, \theta_{t-k}^{loc}\right)$

<u>Compute pose transformation matrix</u>
that can transform vectors from vehicle frame at t-k to vehicle frame at t

$$T_{prev} = \begin{bmatrix} \cos\left(\theta_{t-k}^{loc}\right) & -\sin\left(\theta_{t-k}^{loc}\right) & px_{t-k}^{loc} \\ \sin\left(\theta_{t-k}^{loc}\right) & \cos\left(\theta_{t-k}^{loc}\right) & py_{t-k}^{loc} \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_{curr} = \begin{bmatrix} \cos\left(\theta_{t}^{loc}\right) & -\sin\left(\theta_{t}^{loc}\right) & px_{t}^{loc} \\ \sin\left(\theta_{t}^{loc}\right) & \cos\left(\theta_{t}^{loc}\right) & py_{t}^{loc} \\ 0 & 0 & 1 \end{bmatrix}$$

$$T = T_{curr}^{-1} T_{prev} = \begin{bmatrix} \boldsymbol{R_{2\times 2}} & \boldsymbol{t_{2\times 1}} \\ 0_{1\times 2} & 1 \end{bmatrix}$$

$R_{2\times 2}$

$t_{2\times 1}$

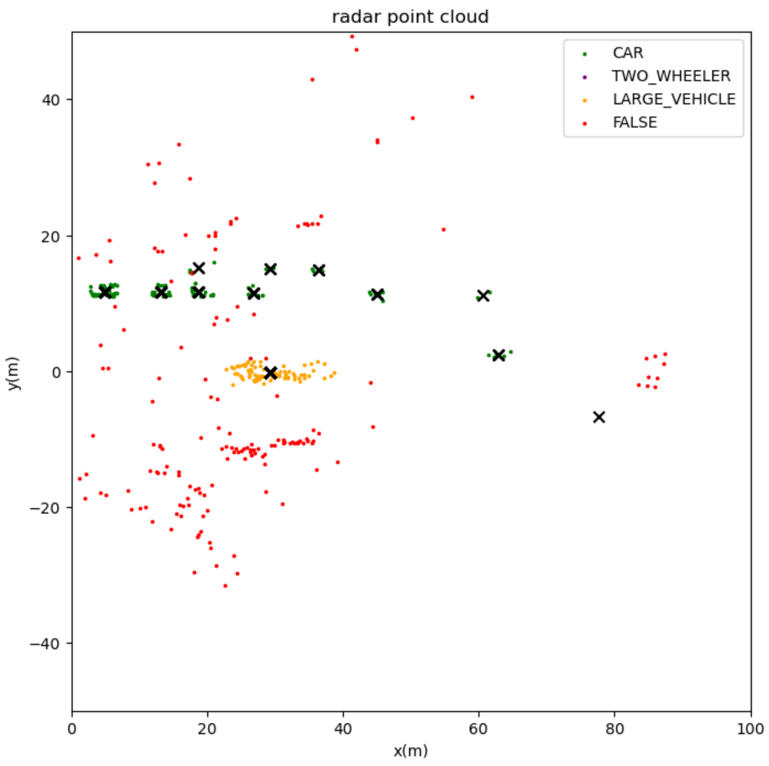
<u>input</u>: radar measurements at t-k:
$\left(x_{t-k}^i, y_{t-k}^i\right)$

<u>Ego motion compensation</u>

$$\begin{pmatrix} x_t^i \\ y_t^i \end{pmatrix} = R_{2\times 2} \begin{pmatrix} x_{t-k}^i \\ y_{t-1}^i \end{pmatrix} + t_{2\times 1}$$

<u>output</u>: temporally aligned radar measurements:
$\left(\boldsymbol{x_t^i}, \boldsymbol{y_t^i}\right)$

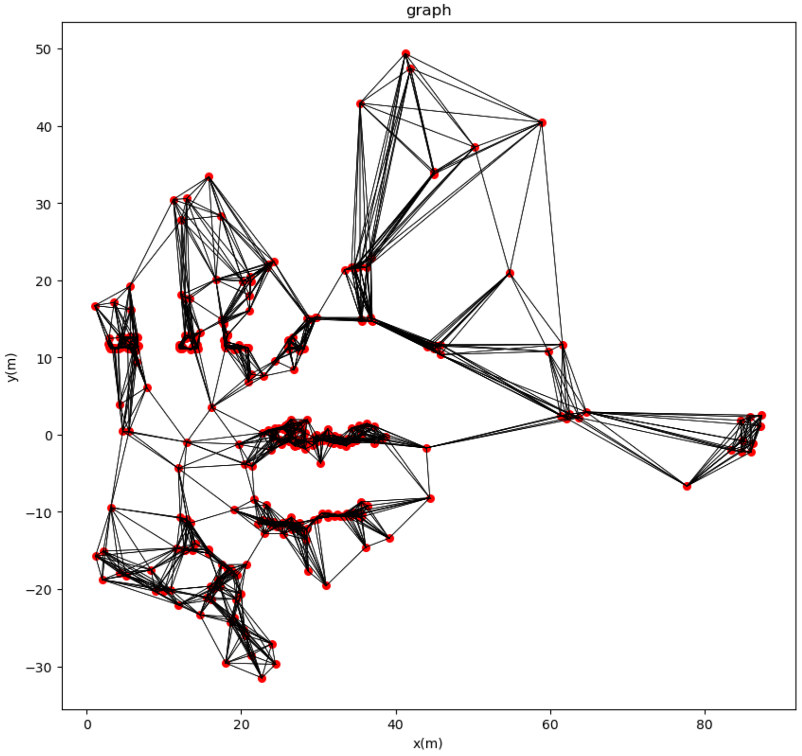
Graph Construction: Adjacency list



Create an **undirected graph** such that **each node** which are the measurements **are connected to K nearest nodes.**

The distance between nodes is computed as:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$



Node i

Node j

	1	*2*	*3*	*4*	*5*	*6*	*7*	...	*n-1*	*n*
1	0	1	1	1	0	1	0	...	0	1
2	1	0	0	0	1	0	1	...	0	0
3	1	0	0	0	0	0	0	...	0	0
4	1	0	0	0	0	0	0	...	0	0
5	0	1	0	0	0	0	0	...	1	1
6	1	0	0	0	0	0	0	...	0	0
7	0	1	0	1	0	0	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...
n-1	0	0	0	0	1	0	0	...	0	1
n	1	0	0	0	1	0	0	...	1	0

Create an adjacency list representation of the graph edges (required input format for pytorch geometric)

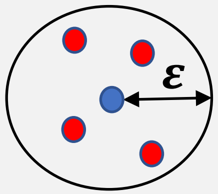
i	*j*
1	*2*
1	*3*
1	*4*
1	*6*
1	*n*
2	*1*
2	*5*
2	*7*
3	*1*
4	*1*
5	*1*
5	*n-1*
5	*n*
6	*1*
7	*2*
7	*4*
...	...

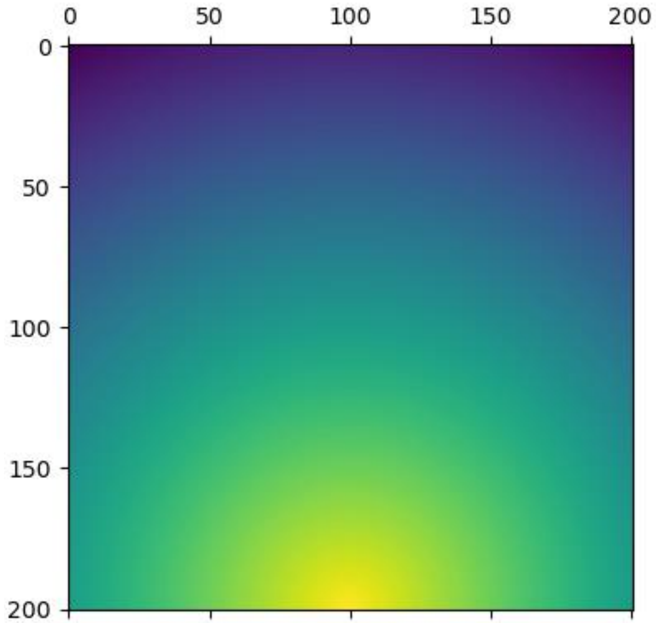
Graph Construction: Node Features

Each of input node feature vector $\boldsymbol{v_i}$ consists of :

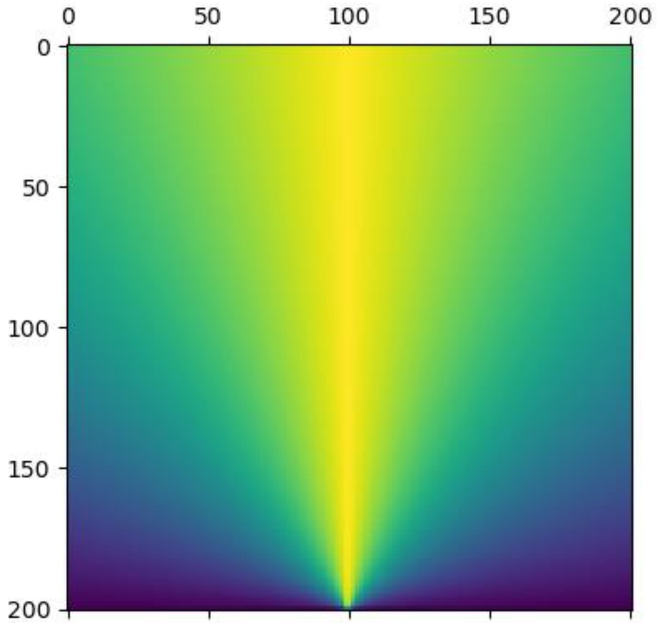
- ***vr*** : range rate
- ***rcs***: radar cross section of the target
- $\boldsymbol{n_{degree}}$: node degree
- ***t***: measurement frame capture time
- $\boldsymbol{conf_{range}}$: measurement range confidence
- $\boldsymbol{conf_{azimuth}}$: measurement azimuth confidence

Node degree of a vertex $\boldsymbol{v_i}$ is the number of nodes that falls with a circle of radius $\boldsymbol{\varepsilon}$ centred at the node $\boldsymbol{v_i}$





$$\mathbf{conf_{range}} = \frac{\mathbf{r_i - r_{max}}}{\mathbf{r_{min} - r_{max}}}$$



$$\mathbf{conf_{azimuth}} = \frac{|\boldsymbol{\theta_i}| - \boldsymbol{\theta_{max}}}{\boldsymbol{\theta_{min}} - \boldsymbol{\theta_{max}}}$$

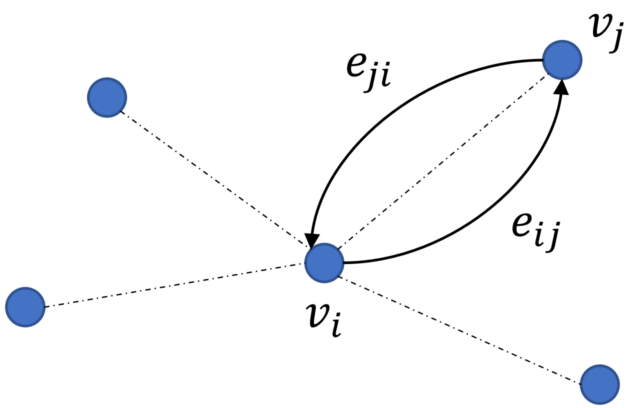
Graph Construction: Edge Features

Each of input edge vector $\mathbf{e_{ij}}$ consists of :

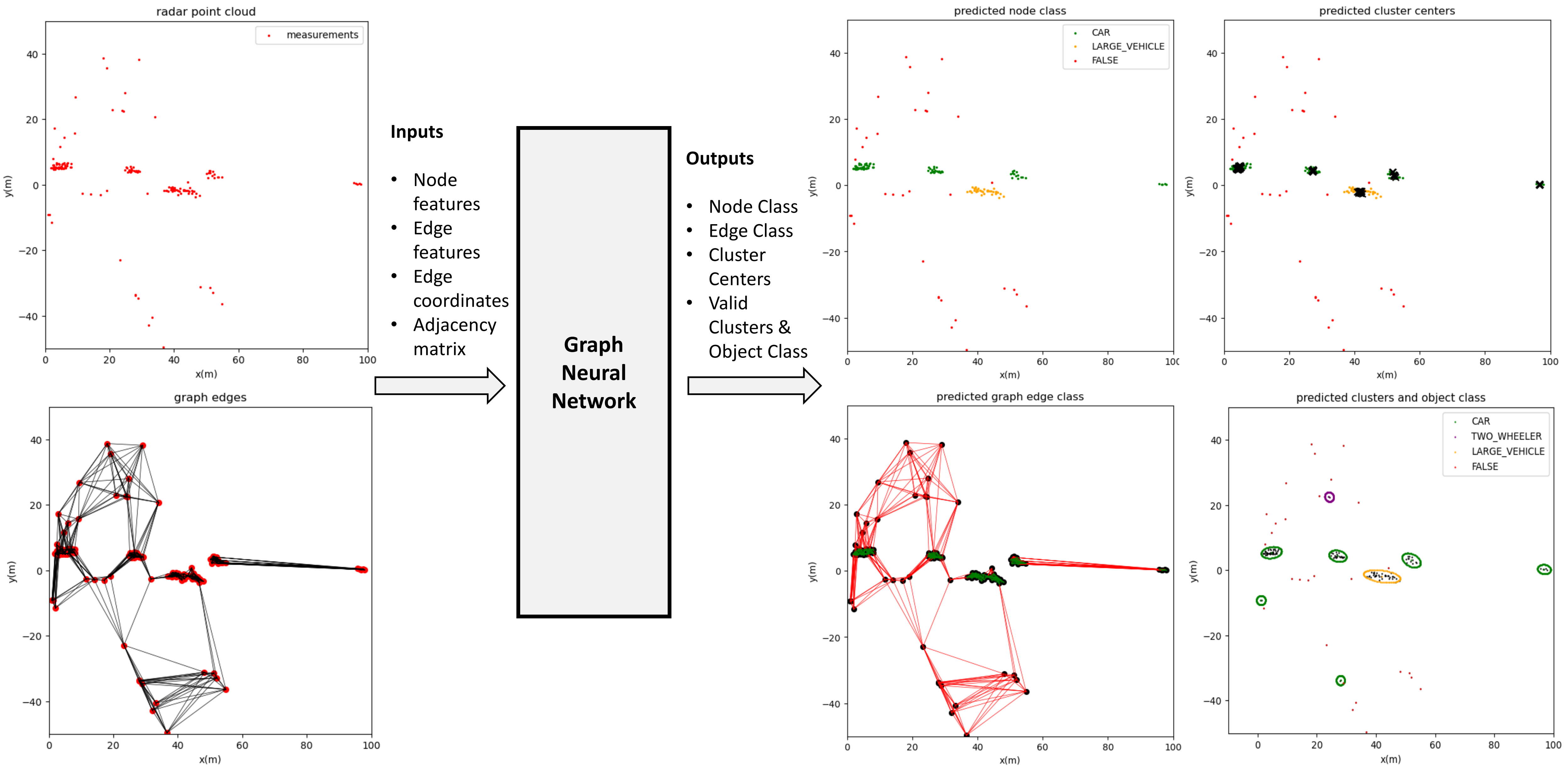
- $\Delta\mathbf{x} = \mathrm{x_i} - \mathrm{x_j}$
- $\Delta\mathbf{y} = \mathrm{y_i} - \mathrm{y_j}$
- $\Delta\mathbf{l} = \sqrt{(\Delta\mathbf{x})^2 + (\Delta\mathbf{y})^2}$
- $\Delta\mathbf{vx} = \mathrm{vx_i} - \mathrm{vx_j}$
- $\Delta\mathbf{vy} = \mathrm{vy_i} - \mathrm{vy_j}$
- $\Delta\mathbf{v} = \sqrt{(\Delta\mathbf{vx})^2 + (\Delta\mathbf{vy})^2}$
- $\Delta\mathbf{t} = \mathrm{t_i} - \mathrm{t_j}$

$v_x = v_r \cos(\theta)$

$v_y = v_r \sin(\theta)$

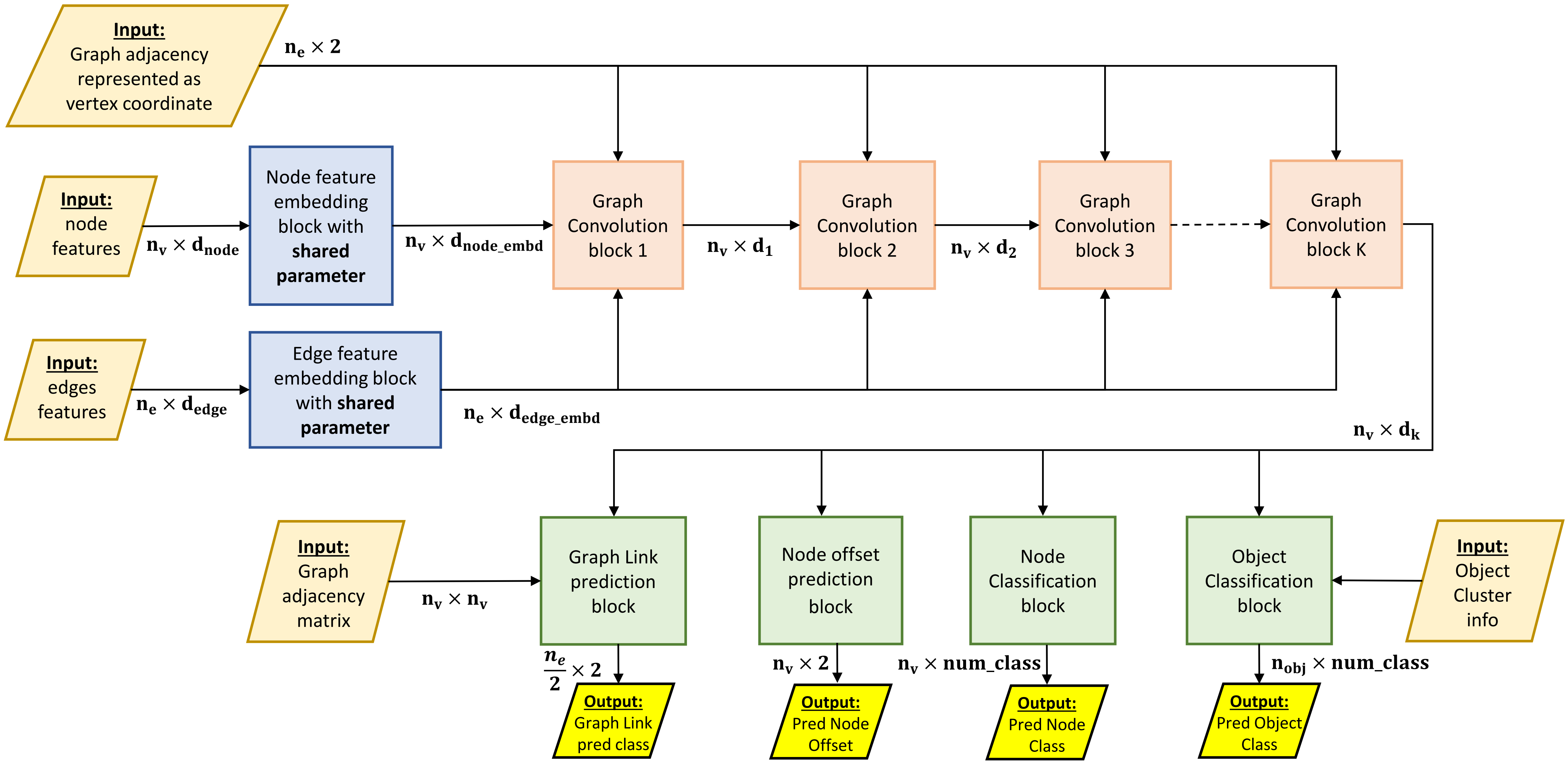


An **undirected edge** can be decomposed as **two directed edges**. Input edge features are computed for all valid directed edges from the adjacency list

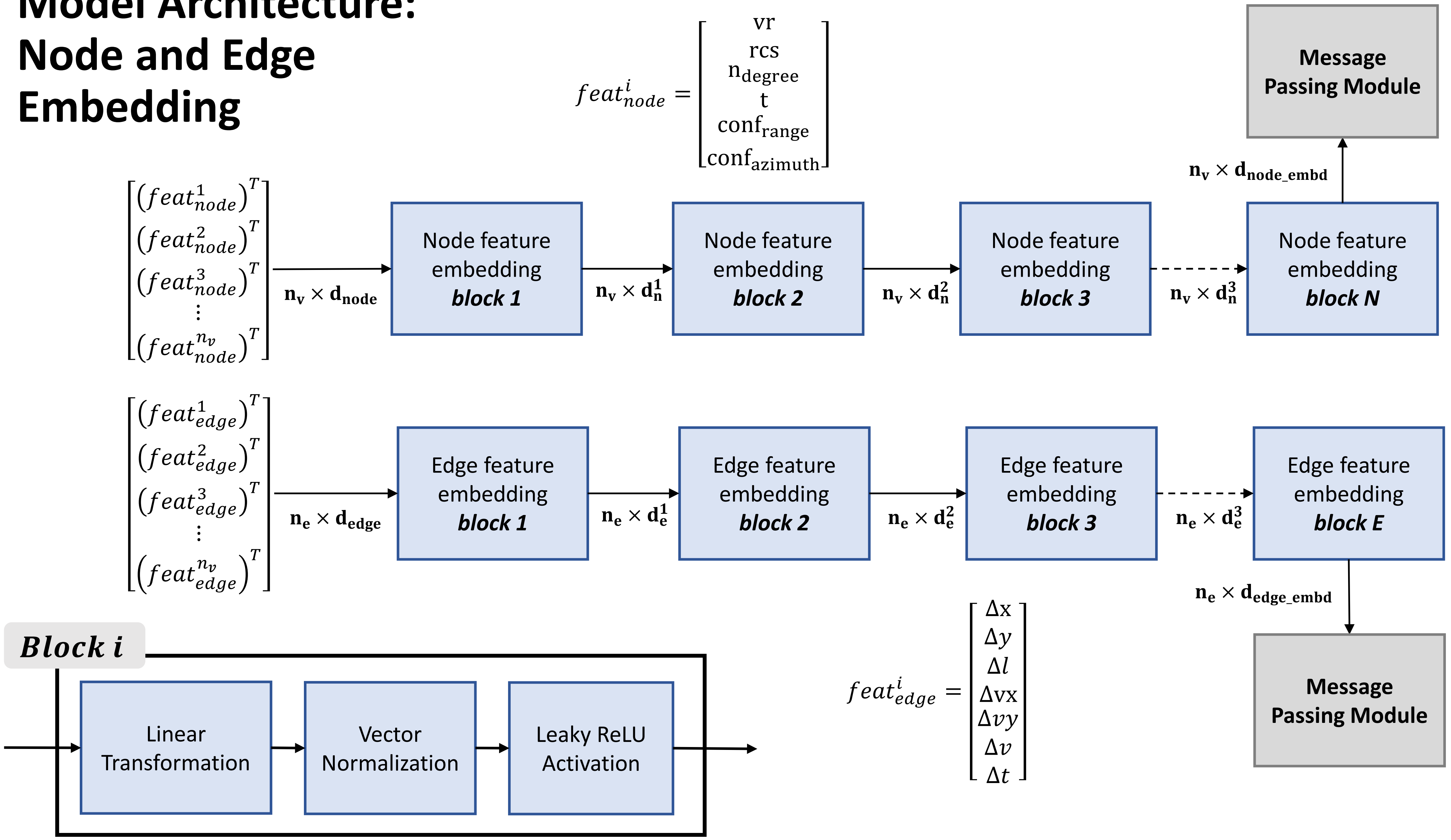


radar point cloud
measurements
graph edges
Inputs
Node features
Edge features
Edge coordinates
Adjacency matrix
Graph Neural Network
Outputs
Node Class
Edge Class
Cluster Centers
Valid Clusters & Object Class
predicted node class
CAR
LARGE_VEHICLE
FALSE
predicted cluster centers
predicted graph edge class
predicted clusters and object class
CAR
TWO_WHEELER
LARGE_VEHICLE
FALSE
x(m)
y(m)

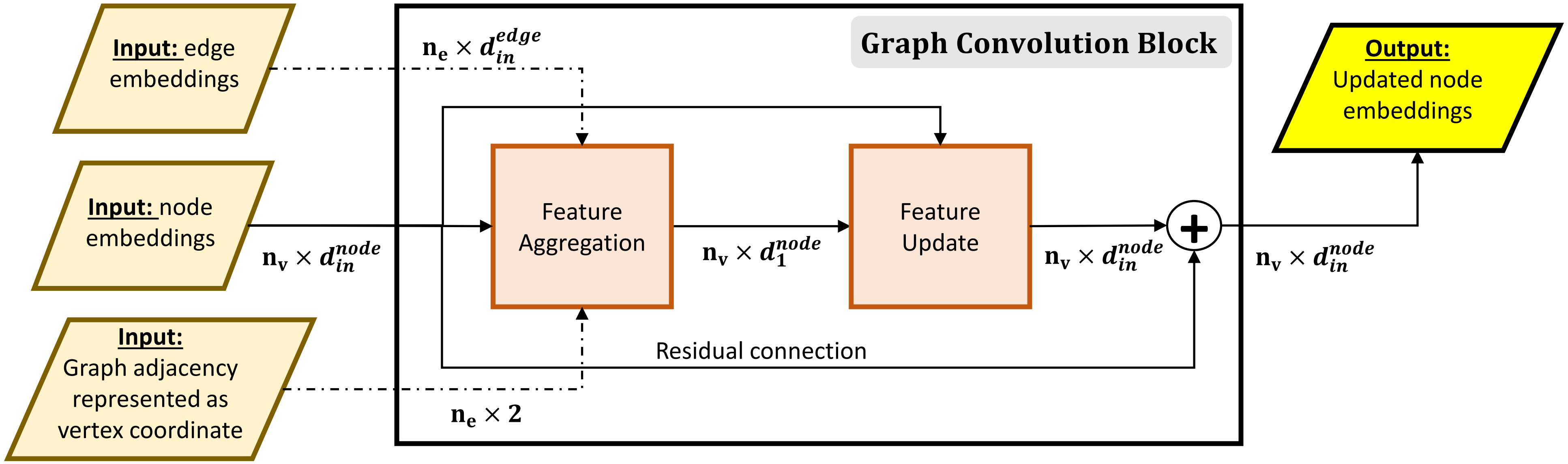
Model Architecture: High Level

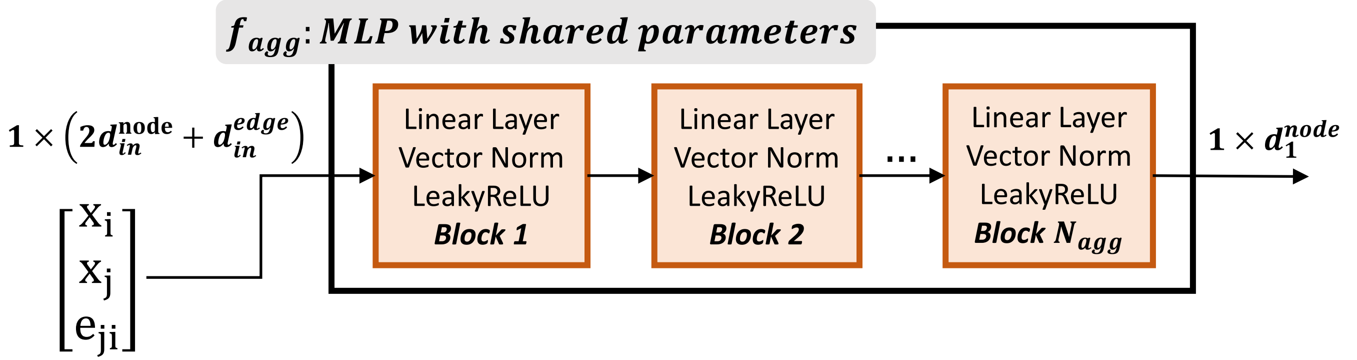


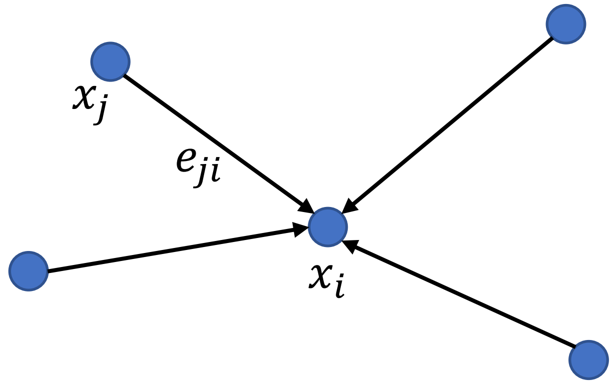
Model Architecture: Node and Edge Embedding

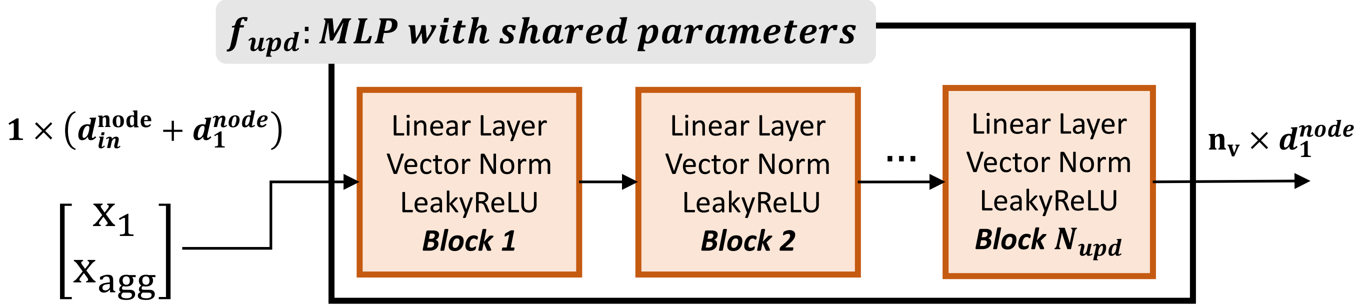


Model Architecture: Graph Convolution



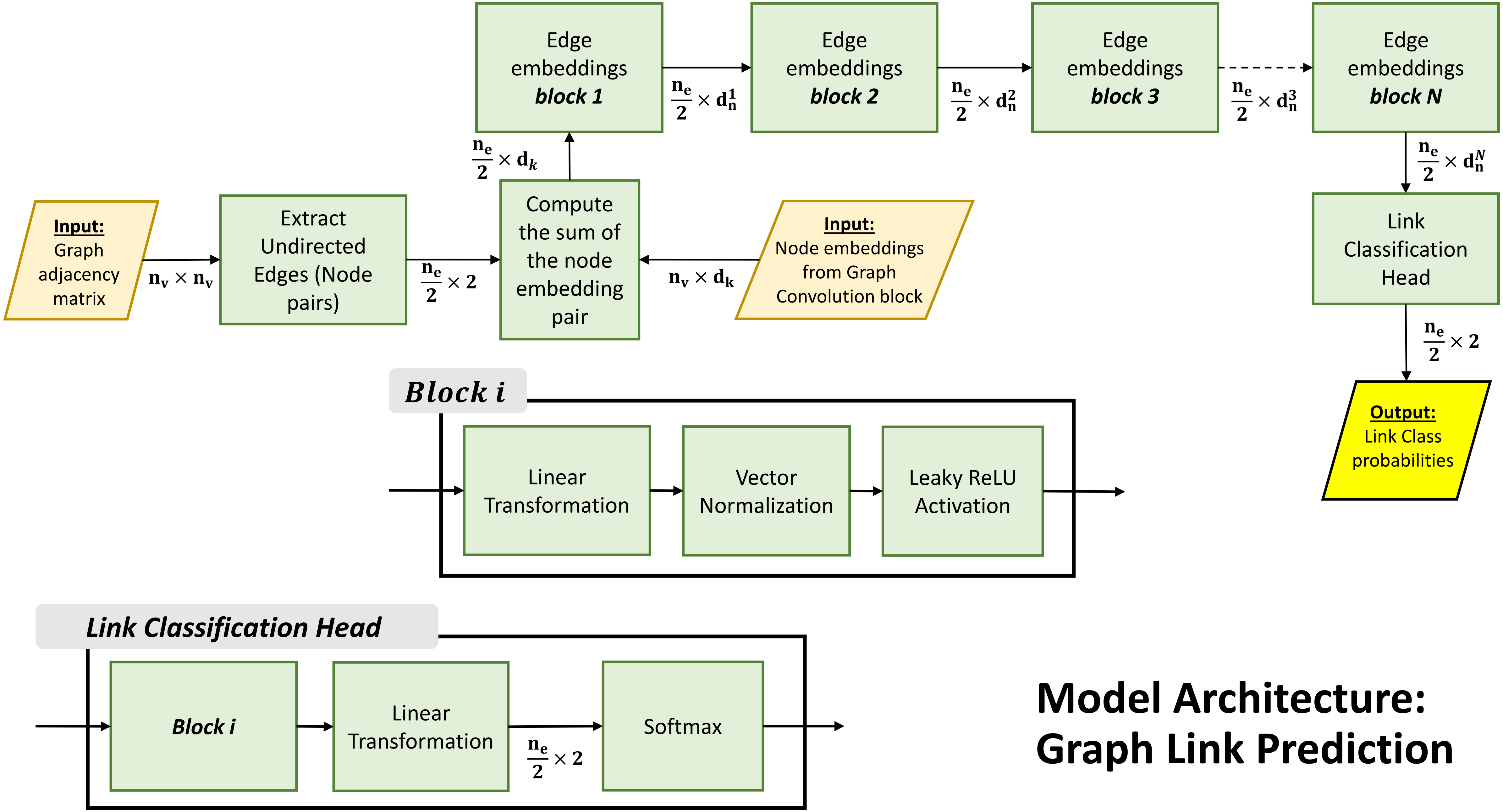


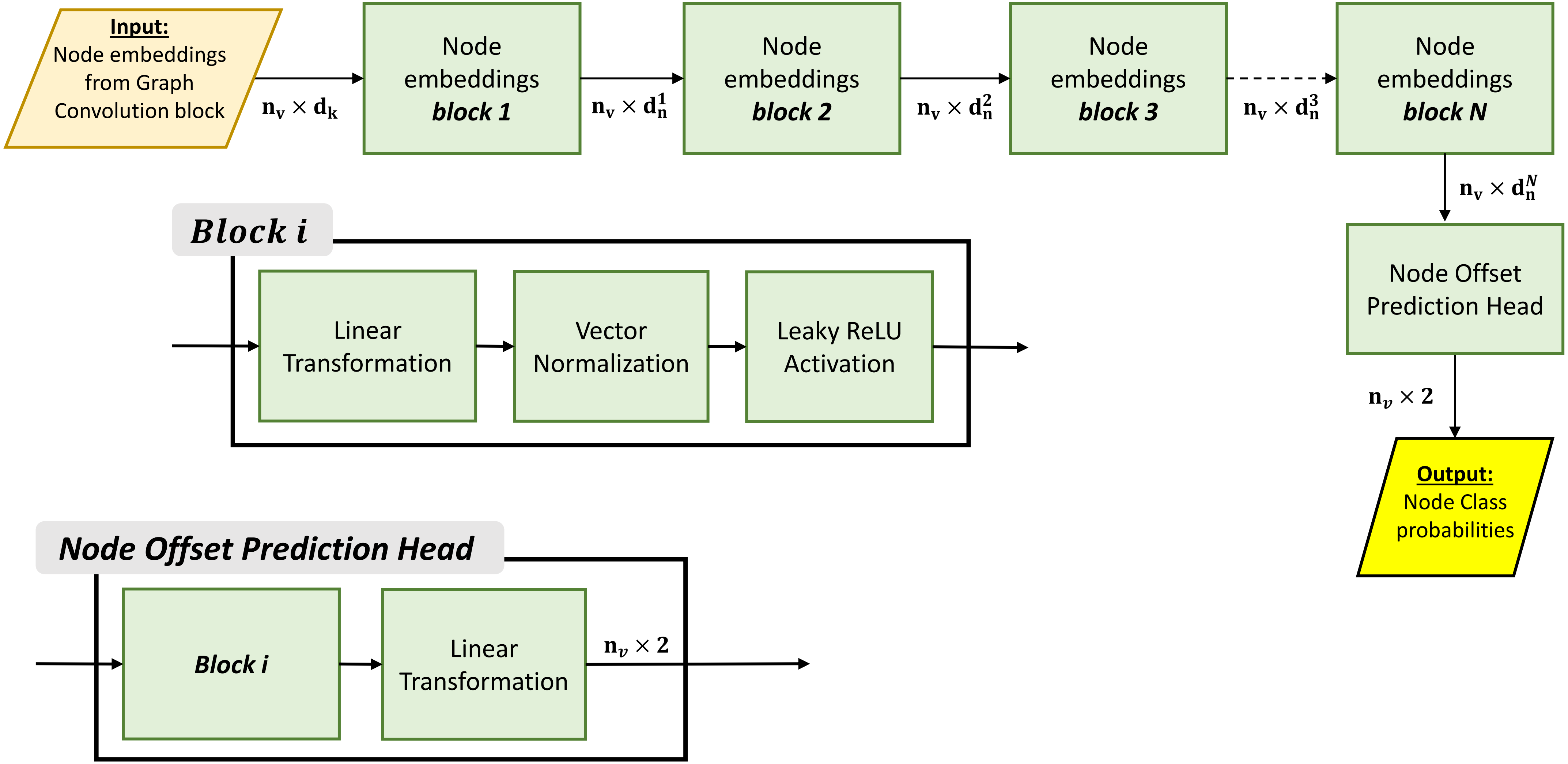




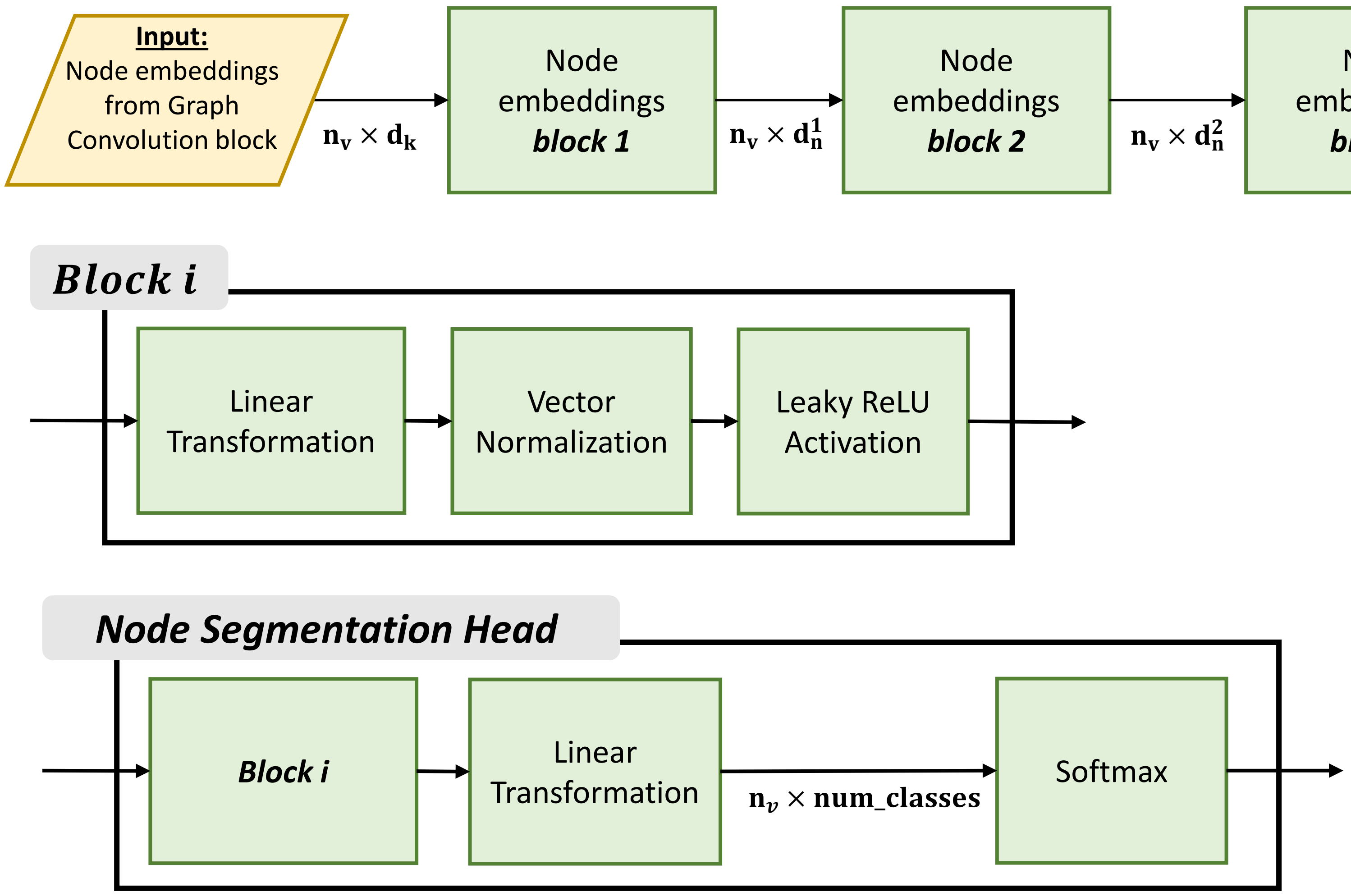
$$\mathrm{x_{agg}} = \sum_{j \in \mathcal{N}(i)} \mathrm{f_{agg}}(\mathrm{x}_{concat}) \quad \text{where, } \mathrm{x_{concat}} = [\mathrm{x_i}, \mathrm{x_j}, \mathrm{e_{ji}}]$$

$$\mathrm{x_{upd}} = \mathrm{f_{upd}}(\mathrm{y_{concat}}) \quad \text{where, } \mathrm{y_{concat}} = [\mathrm{x_i}, \mathrm{x_{agg}}]$$

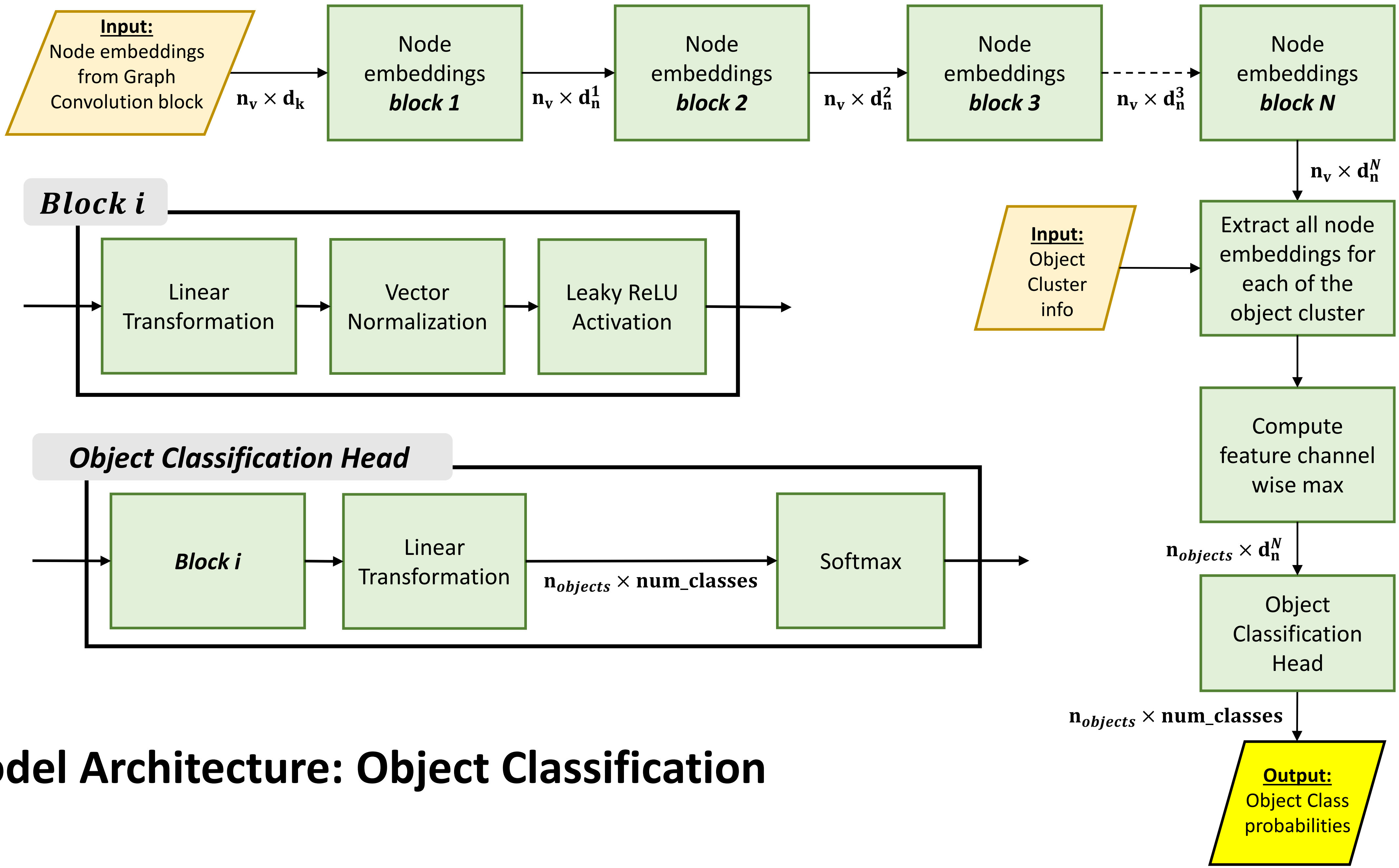

Input: Graph adjacency matrix
$n_v \times n_v$
Extract Undirected Edges (Node pairs)
$\frac{n_e}{2} \times 2$
Compute the sum of the node embedding pair
$n_v \times d_k$
Input: Node embeddings from Graph Convolution block
$\frac{n_e}{2} \times d_k$
Edge embeddings block 1
$\frac{n_e}{2} \times d_n^1$
Edge embeddings block 2
$\frac{n_e}{2} \times d_n^2$
Edge embeddings block 3
$\frac{n_e}{2} \times d_n^3$
Edge embeddings block N
$\frac{n_e}{2} \times d_n^N$
Link Classification Head
$\frac{n_e}{2} \times 2$
Output: Link Class probabilities
Block i
Linear Transformation
Vector Normalization
Leaky ReLU Activation
Link Classification Head
Block i
Linear Transformation
$\frac{n_e}{2} \times 2$
Softmax
Model Architecture: Graph Link Prediction



Model Architecture: Node Offset Prediction



Model Architecture: Node Segmentation



Model Architecture: Object Classification