ILC/Particle Flow Algorithm using Graph Neural Network

SHUSAKU TSUMURA/KYUSHU UNIVERSITY

Papers

- GravNet :
  Learning representations of irregular particle-detector geometry with distance-weighted graph networks
  Eur. Phys. J. C (2019) 79:608
  https://doi.org/10.1140/epjc/s10052-019-7113-9
- Object condensation :
  Object condensation: one-stage grid-free multi-object reconstruction in physics detectors, graph, and image data
  Eur. Phys. J. C (2020) 80:886
  https://doi.org/10.1140/epjc/s10052-020-08461-2
- Whole approach :
  Multi-particle reconstruction in the High Granularity Calorimeter using object condensation and graph neural networks
  https://arxiv.org/abs/2106.01832

GravNet - Network -

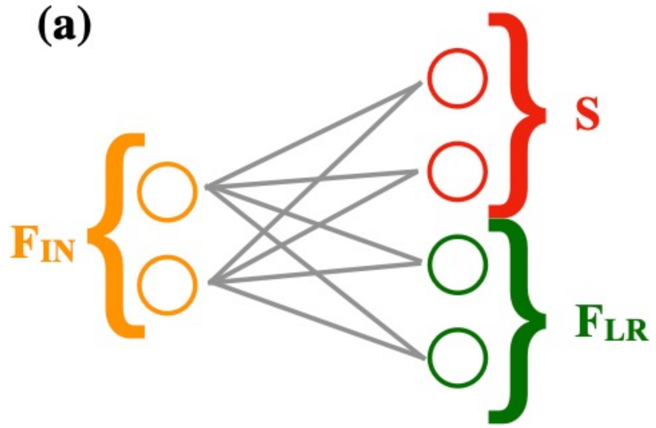
- Input Data : $B \times V \times F_{IN}$

  B : Number of examples including in a batch

  V : Number of hits for each detector

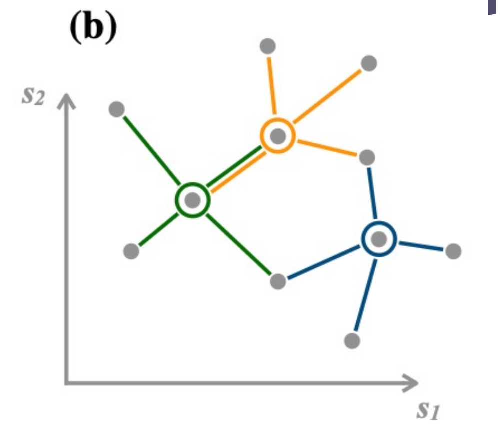
  F_{IN} : Number of the features for each hit

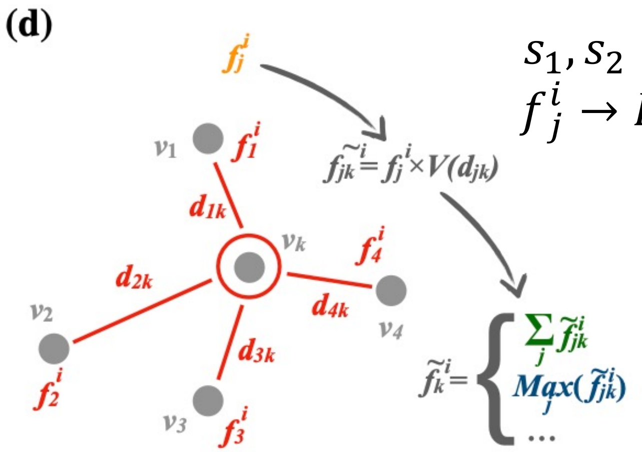
- S : Set of coordinates in some learned representation space
- F_{LR} : learned representation of the vertex features



GravNet

- Input example of initial dimension $V \times F_{IN}$ is converted into a graph.
- the f_j^i features of the v_j vertices connected to a given vertex or aggregator v_k are converted into the $\widetilde{f_{jk}}^i$ quantities, through a potential (function of euclidean distance d_{jk}).
- The potential function $V(d_{jk})$ is introduced to enhance the contribution of close-by vertices.
  Example: $V(d_{jk}) = \exp(-d_{jk}^2)$
- The $\widetilde{f_{jk}}^i$ functions computed from all the edges associated to a vertex of aggregator v_k are combined, generating a new feature $\widetilde{f_k}^i$ of v_k.
  Example : the average of the $\widetilde{f_{jk}}^i$ across the j edges / their maximum



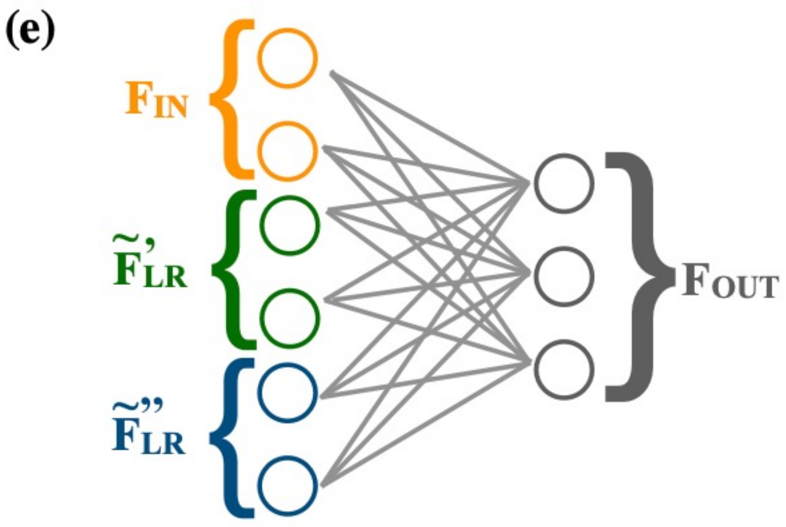


$s_1, s_2 \rightarrow S$

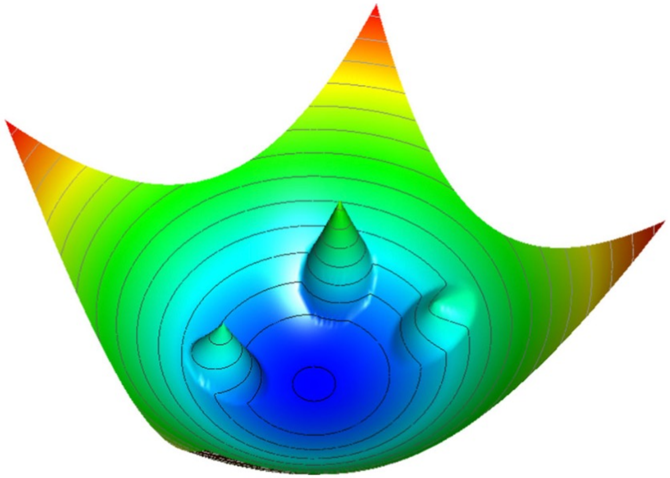
$f_j^i \rightarrow F_{LR}$

GravNet

- For each choice of gathering function, a new set of features $\widetilde{f_k}^{i} \in \widetilde{F_{LR}}$ is generated.
- The $\widetilde{F_{LR}}$ vector is concatenated to the initial vector.
- Activation function : tanh
- The F_{OUT} output carries collective information from each vertex and its surrounding.



Loss function - Network Learning -



- The object condensation approach :
  Aiming to accumulate all object properties in condensation points

Assignment of vertices for each sower	Identification of noise	Update of loss term

- The value of β_i $(0 < \beta_i < 1)$ is used to define a charge q_i per vertex i

$$q_i = \operatorname{arctanh}^2 \beta_i + q_{\min} \quad (\beta_i \to 1 : q_i \to +\infty)$$

- The charge q_i of each vertex belonging to an object k defines a potential $V_{ik}(x) \propto q_i$
- The force affecting vertex j can be described by $q_j \cdot \nabla V_k(x_j) = q_j \nabla \sum_{i=1}^{N} M_{ik} V_{ik}(x_j, q_i)$

$$M_{ik} = \begin{cases} 1 \; (vertex\ i\ belonging\ to\ object\ k) \\ 0 \; (otherwise) \end{cases}$$

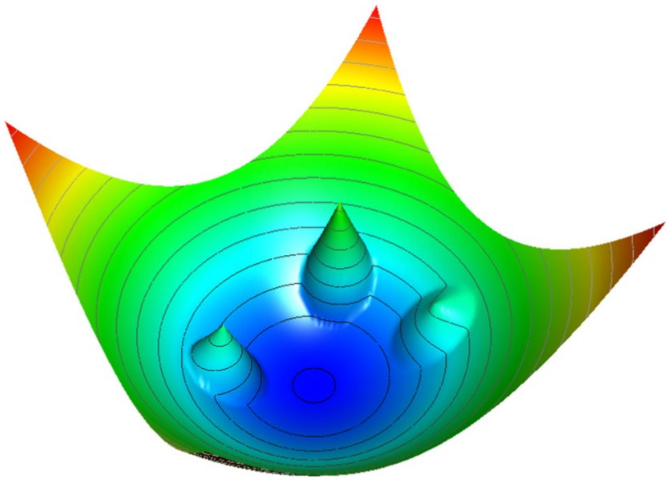
Loss function

- The potential of object k can be approximated :

$$V_k(x) \approx V_{\alpha k}(x, q_{\alpha k}), \quad \text{with } q_{\alpha k} = \max_i q_i M_{ik}.$$

- An attractive and repulsive potential are defined as :

$$\check{V}_k(x) = \|x - x_\alpha\|^2 q_{\alpha k}, \text{ and}$$
$$\hat{V}_k(x) = \max(0, 1 - \|x - x_\alpha\|) q_{\alpha k}.$$



- The total potential loss L_V :

$$L_V = \frac{1}{N} \sum_{j=1}^{N} q_j \sum_{k=1}^{K} \left(M_{jk} \check{V}_k(x_j) + (1 - M_{jk}) \hat{V}_k(x_j) \right)$$

Loss function

- The L_V has the minimum value for $q_i = q_{\min} + \epsilon \ \forall i$
- To enforce one condensation point per object, and none for background or noise vertices, the following additional loss term L_β is introduced :

$$L_\beta = \frac{1}{K}\sum_k (1-\beta_{\alpha k}) + s_B \frac{1}{N_B}\sum_i^N n_i \beta_i,$$

s_B : hyperparameter describing the background suppression strength
K : Maximum value of objects
N_B : Number of background
n_i : Noise tag (if noise, it equals 1.)

- The loss terms are also weighted by $\mathrm{arctanh}^2\, \beta_i$:

$$L_p = \frac{1}{\sum_{i=1}^N \xi_i} \cdot \sum_{i=1}^N L_i(t_i, p_i)\, \xi_i, \text{ with}$$
$$\xi_i = (1-n_i)\,\mathrm{arctanh}^2\, \beta_i .$$

p_i: Featutes
$L_i(t_i, p_i)$: Loss term (Difference between true labels and outputs of network)

Loss function

- If high efficiency instead of high purity is required :

$$L'_p = \frac{1}{K}\sum_{k=1}^{K} \frac{1}{\sum_{i=1}^{N} M_{ik}\xi_i} \cdot \sum_{i=1}^{N} M_{ik} L_i(t_i, p_i)\xi_i .$$

- In practice, individual loss terms might need to be weighted differently :

$$L = L_p + s_C(L_\beta + L_V)$$

Update of loss term

Identification of noise

Assignment of vertices for each sower