VAAD'AIR

Graph Attention Networks

Nicolas Beuve
Nicolas.Beuve@insa-rennes.fr

INSA

IETR

VAADER

Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

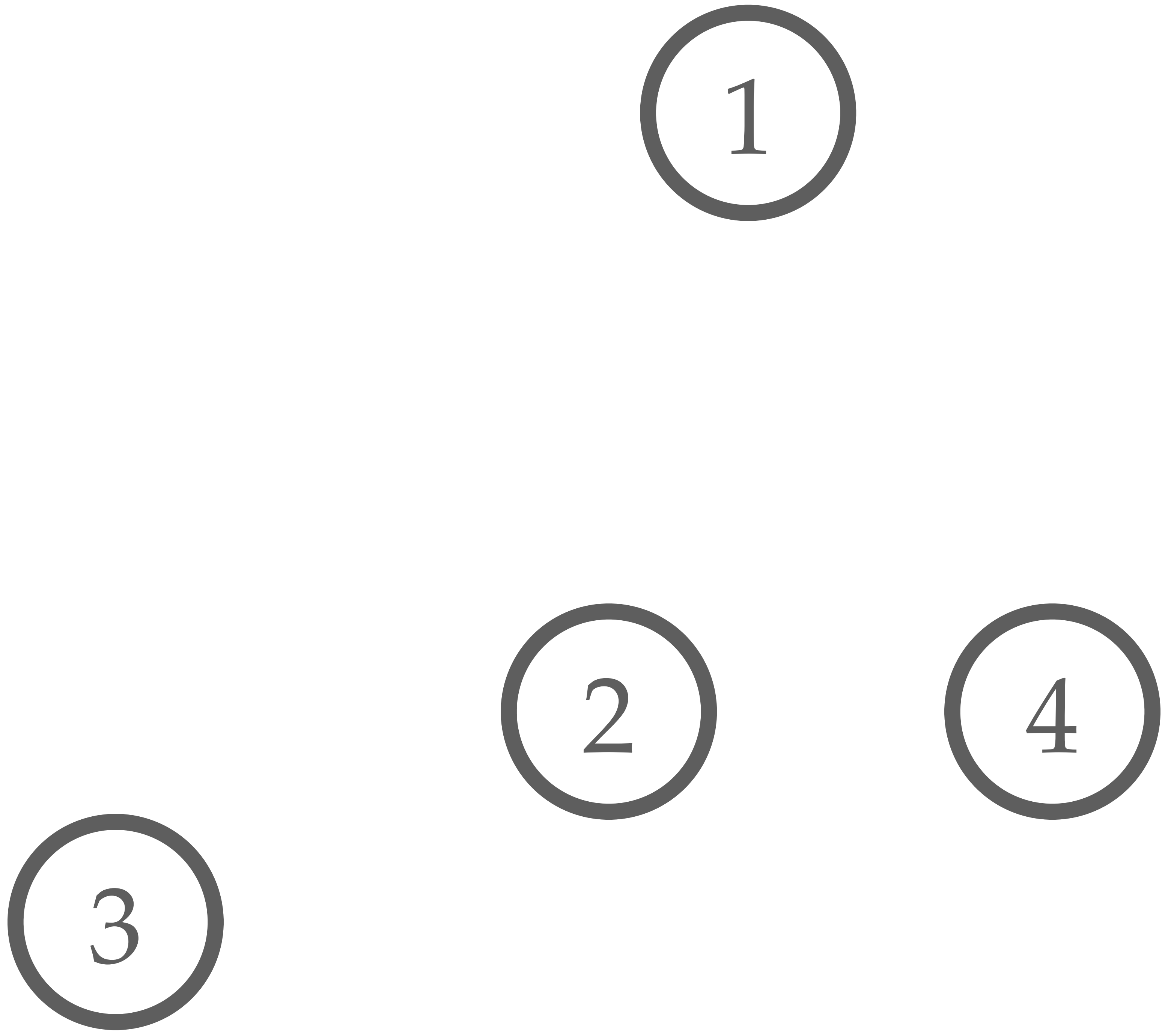
$$\mathbf{N}_i = \{n_j,\ (n_i, n_j) \in \mathbf{E}\}$$

Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

$$\mathbf{N}_i = \{n_j,\ (n_i, n_j) \in \mathbf{E}\}$$

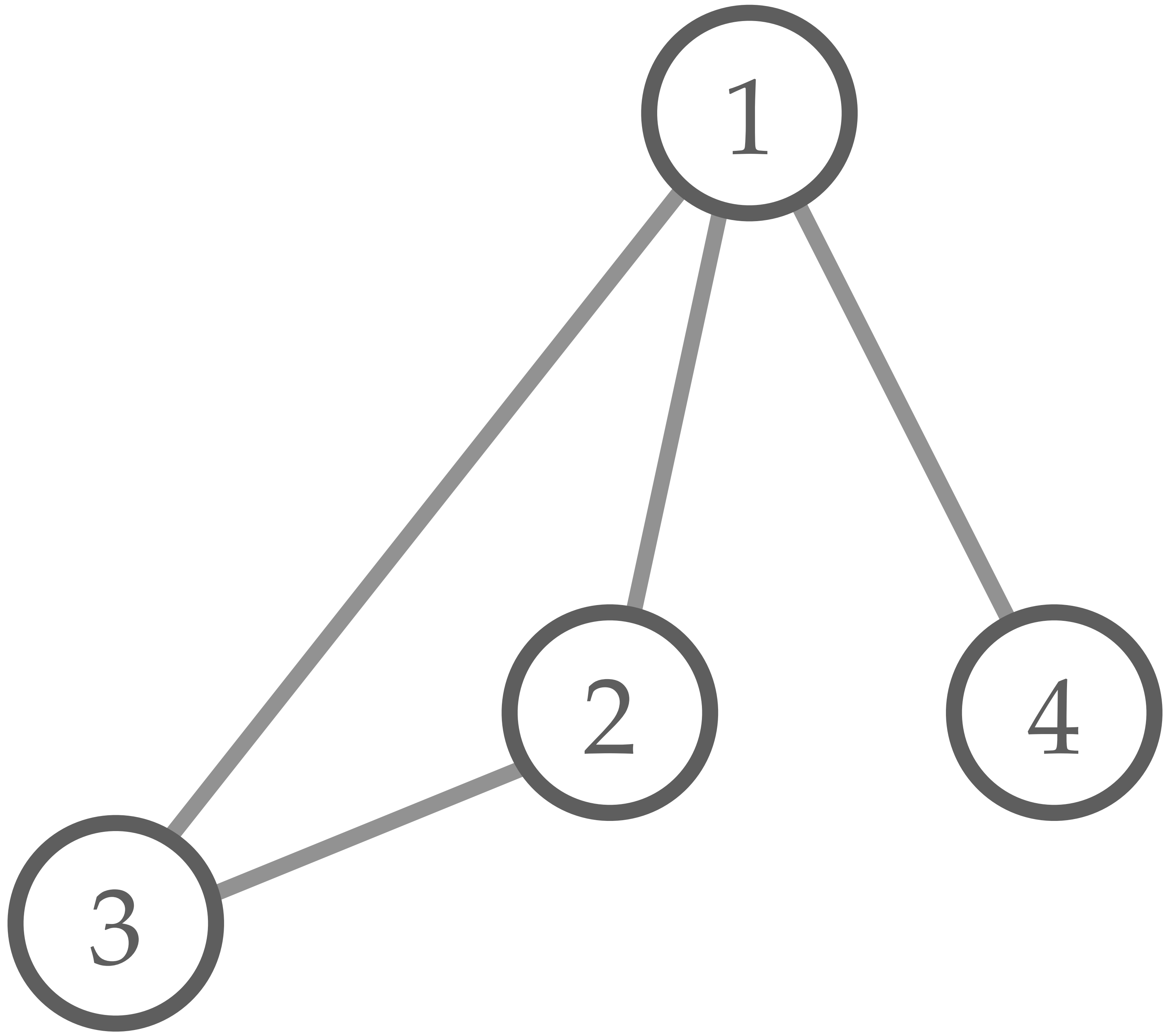


Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

$$\mathbf{N}_i = \{n_j, \ (n_i, n_j) \in \mathbf{E}\}$$

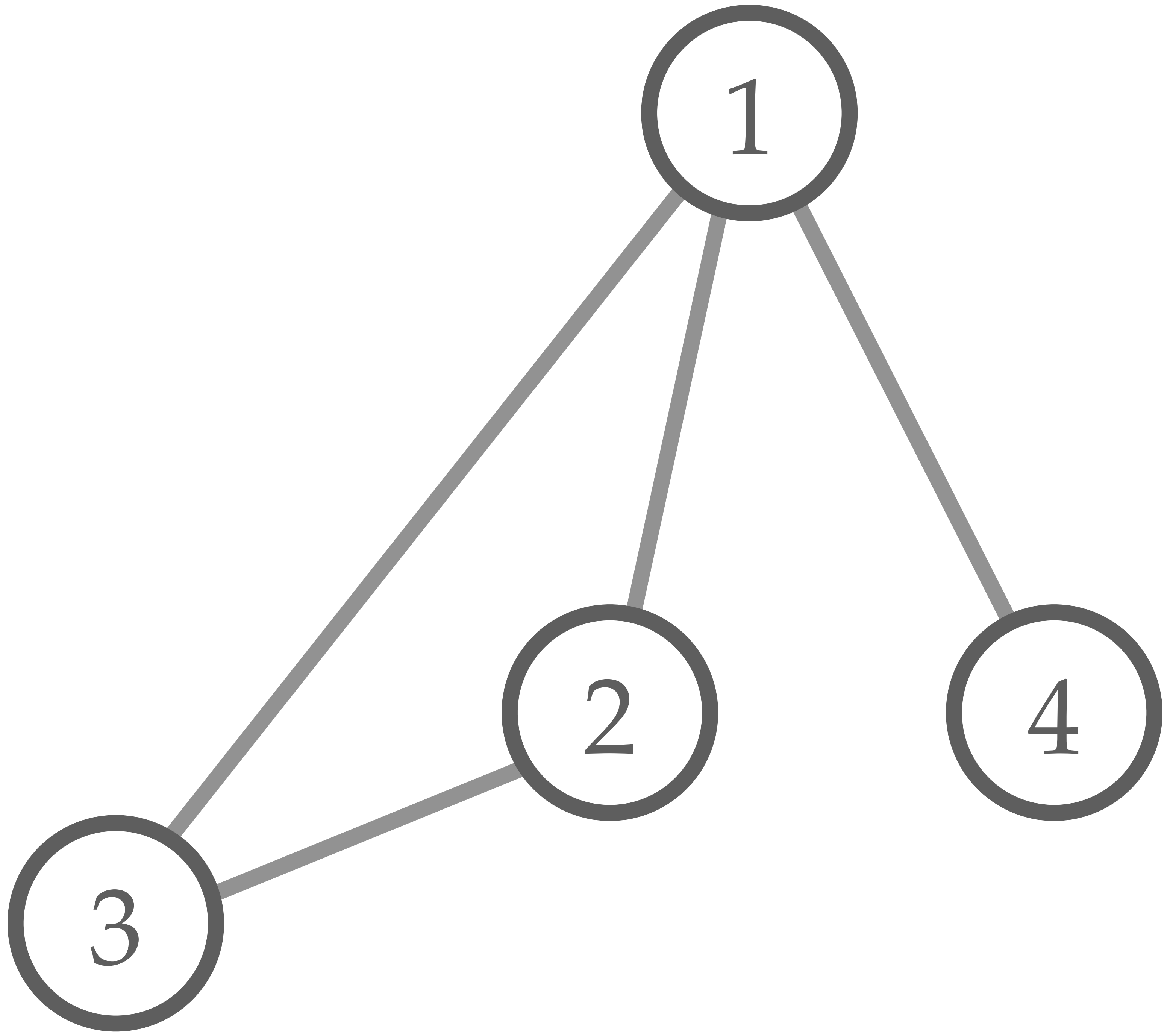


Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

$$\mathbf{N}_i = \{n_j,\ (n_i, n_j) \in \mathbf{E}\}$$

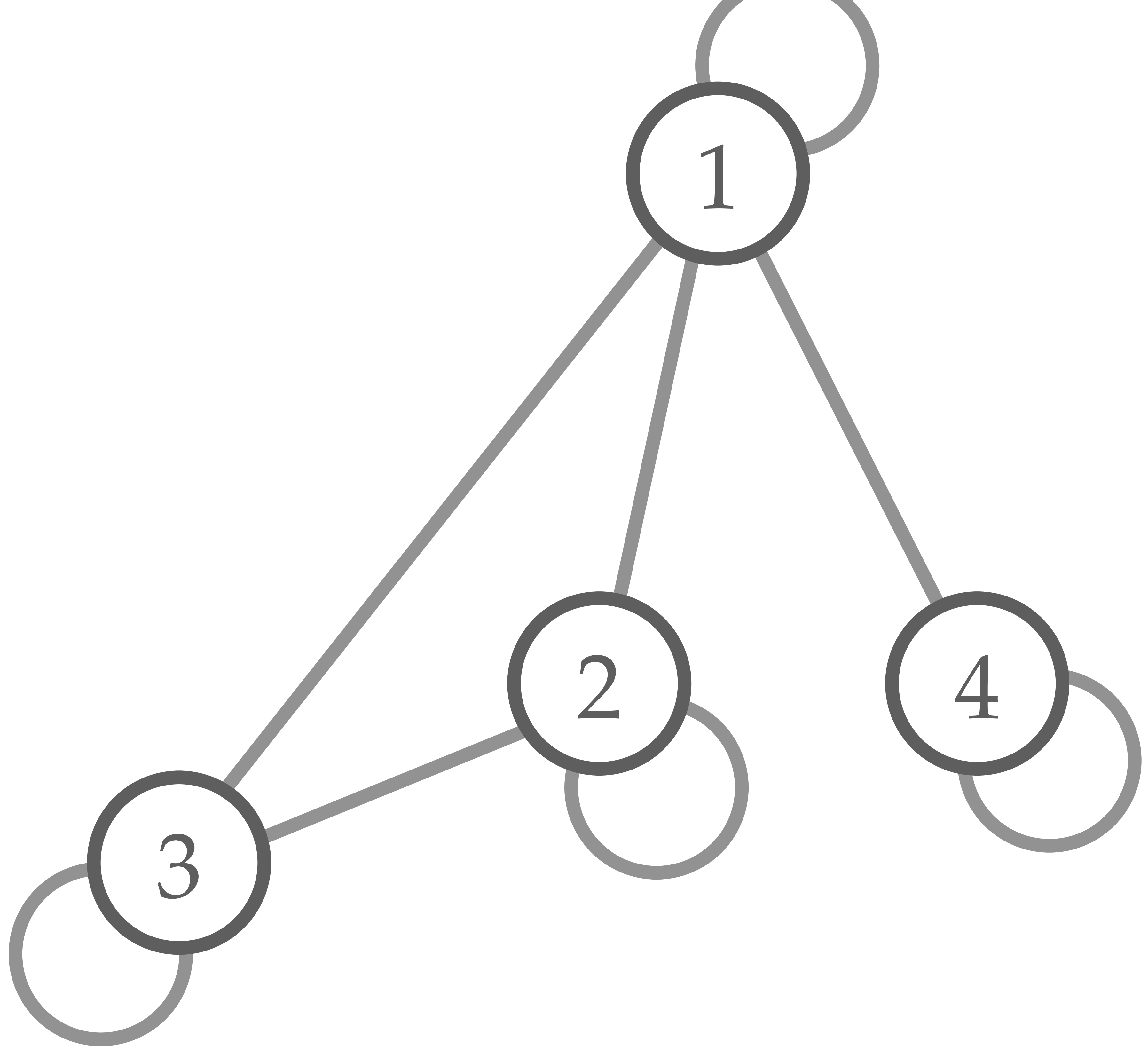


Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

$$\mathbf{N}_i = \{n_j, \ (n_i, n_j) \in \mathbf{E}\}$$

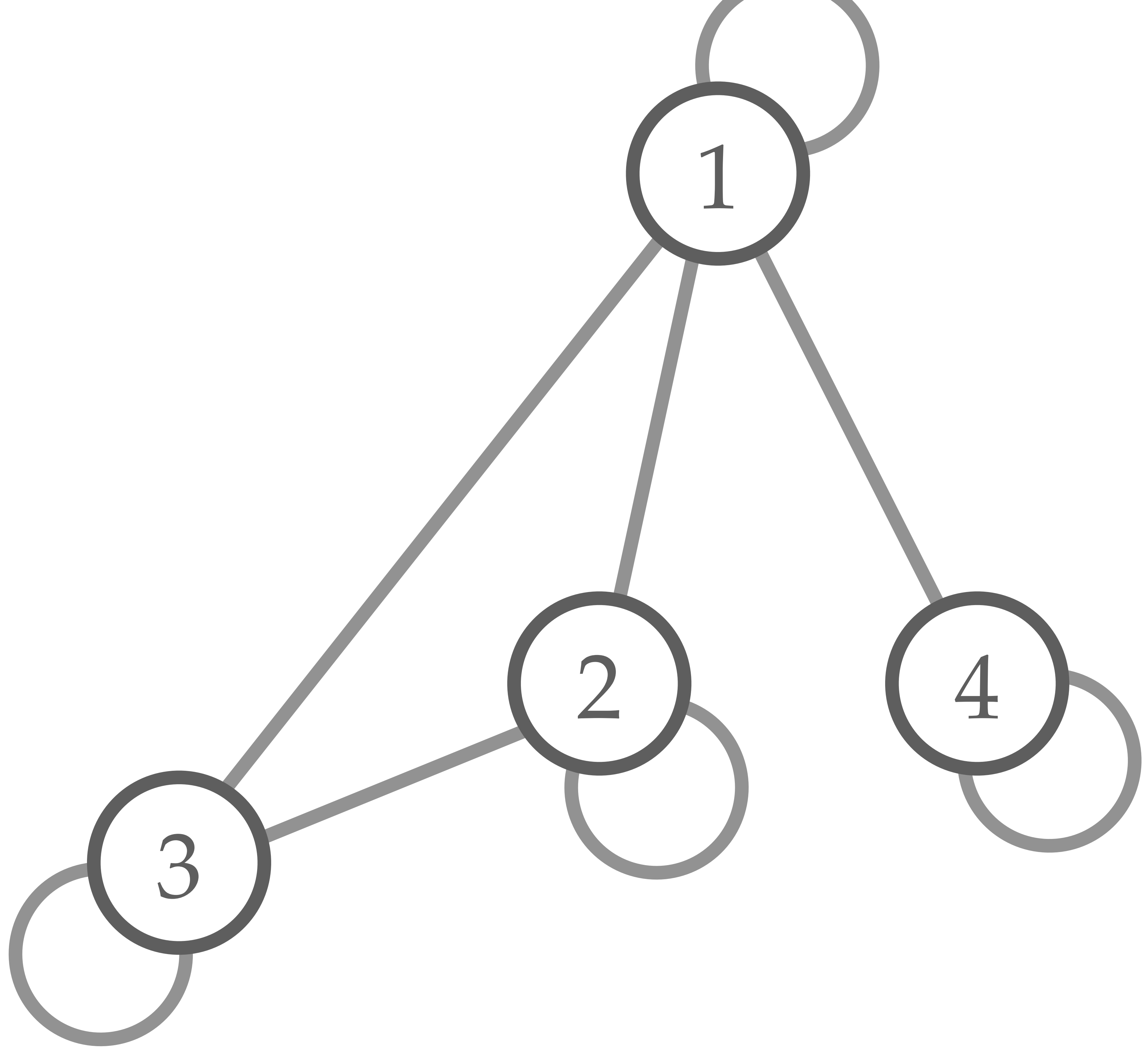


Graph basics & notations

$$\mathbf{G} = (\mathbf{N}, \mathbf{E}) \quad \begin{cases} n = card(N) \\ m = card(E) \end{cases}$$

$$\mathbf{h} = \{\overrightarrow{h}_1, \ldots, \overrightarrow{h}_N\}, \quad \overrightarrow{h}_i \in \mathbb{R}^F$$

$$\mathbf{N}_i = \{n_j, \ (n_i, n_j) \in \mathbf{E}\}$$



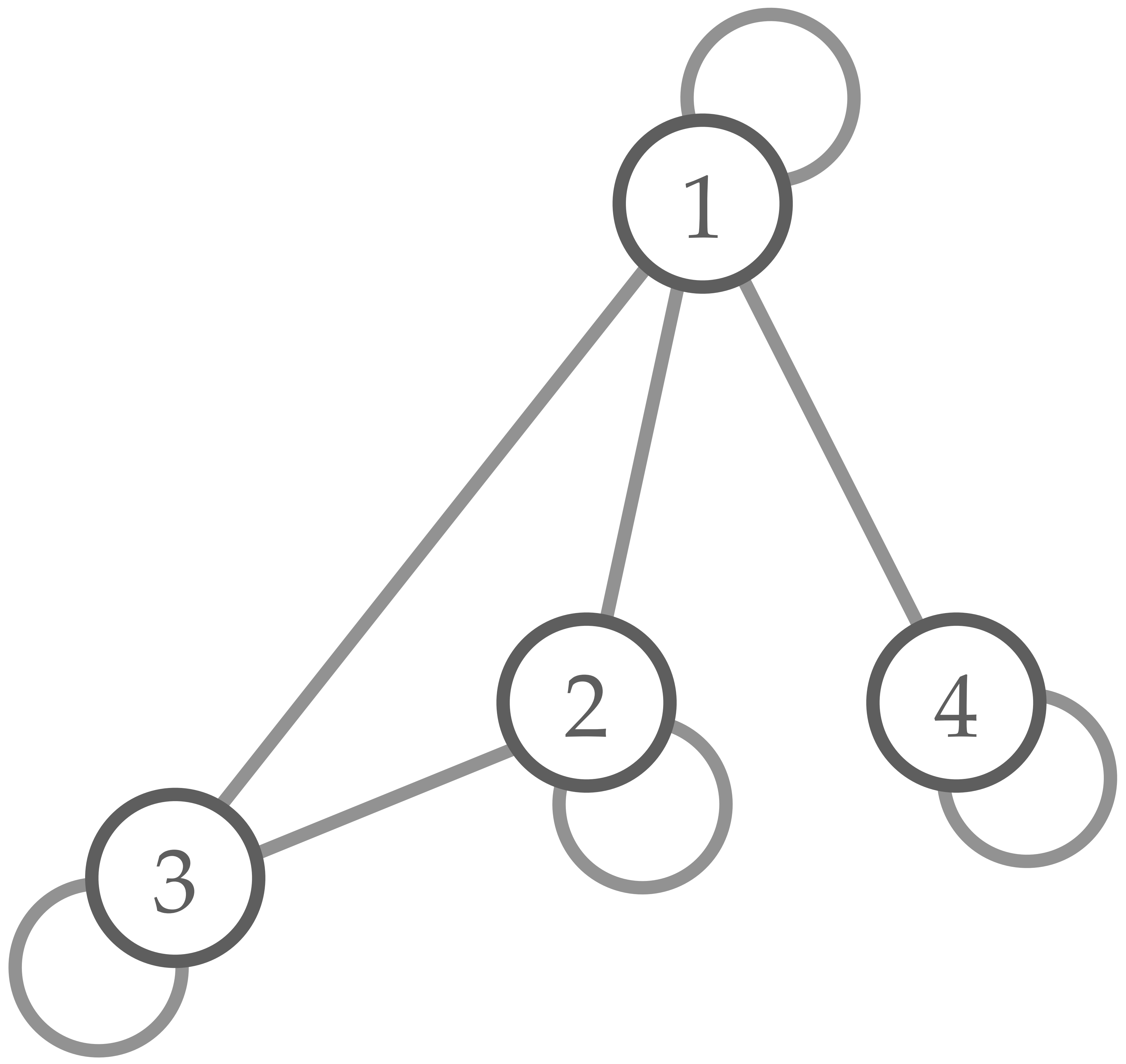
Graph basics & notations

$\mathbf{N} = \{1,2,3,4\}$

$$\mathbf{E} = \{(1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (3,3), (4,4)\}$$

$\mathbf{N}_1 = \{1,2,3,4\}$ $\mathbf{N}_2 = \{1,2,3\}$

$\mathbf{N}_3 = \{1,2,3\}$ $\mathbf{N}_4 = \{1,4\}$



Example of a GNN layer

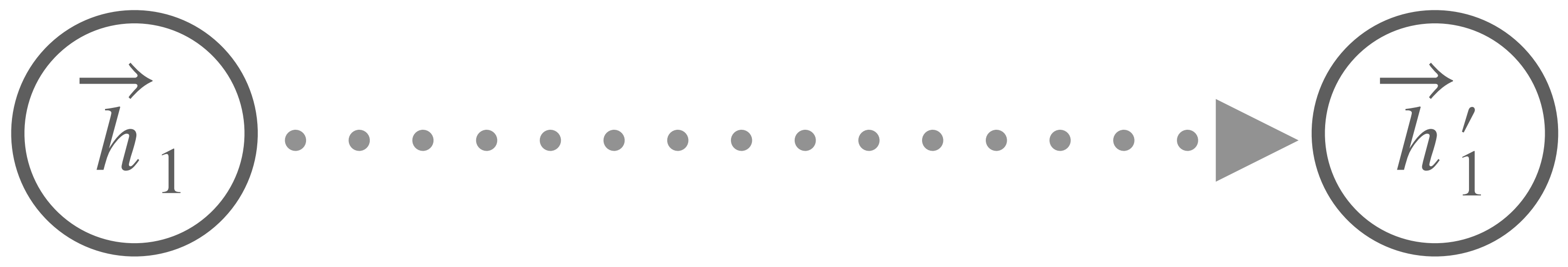
Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h_i' \in \mathbb{R}^{F'}$.

$$\vec{h}_1 \longrightarrow \vec{h}'_1$$

Example of a GNN layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h_i' \in \mathbb{R}^{F'}$.

$$\vec{h}_i' = \sigma\left(\mathbf{W} \vec{h}_i \right)$$

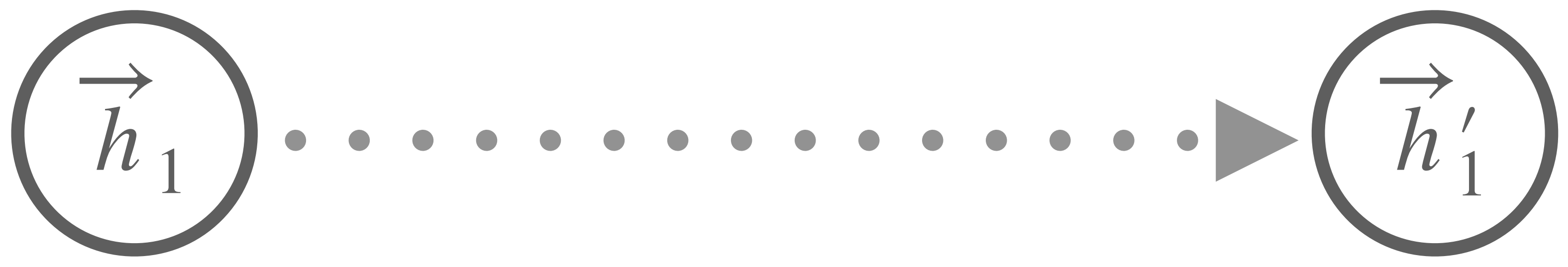


Example of a GNN layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h'_i \in \mathbb{R}^{F'}$.

$$\vec{h}'_i = \sigma\left(\mathbf{W}\,\vec{h}_i\right)$$

Linear transformation $\in \mathbb{R}^{F' \times F}$



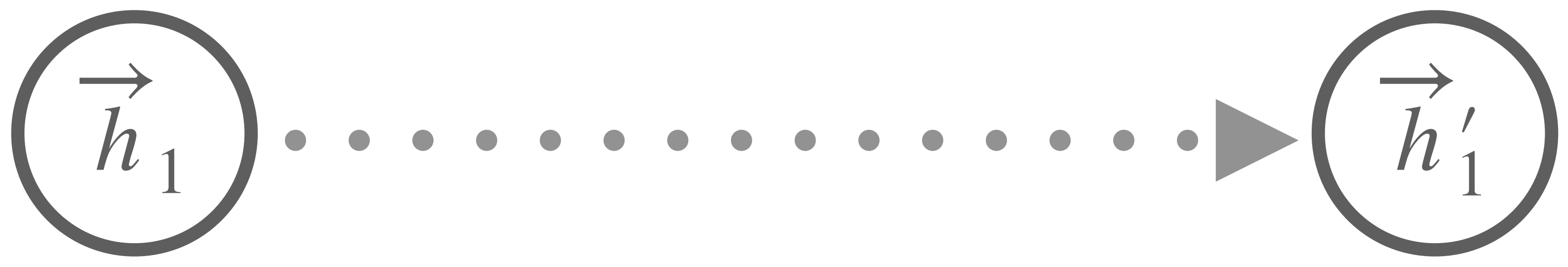
Example of a GNN layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h_i' \in \mathbb{R}^{F'}$.

Activation function

$$\vec{h}_i' = \sigma\left(\mathbf{W}\,\vec{h}_i\right)$$

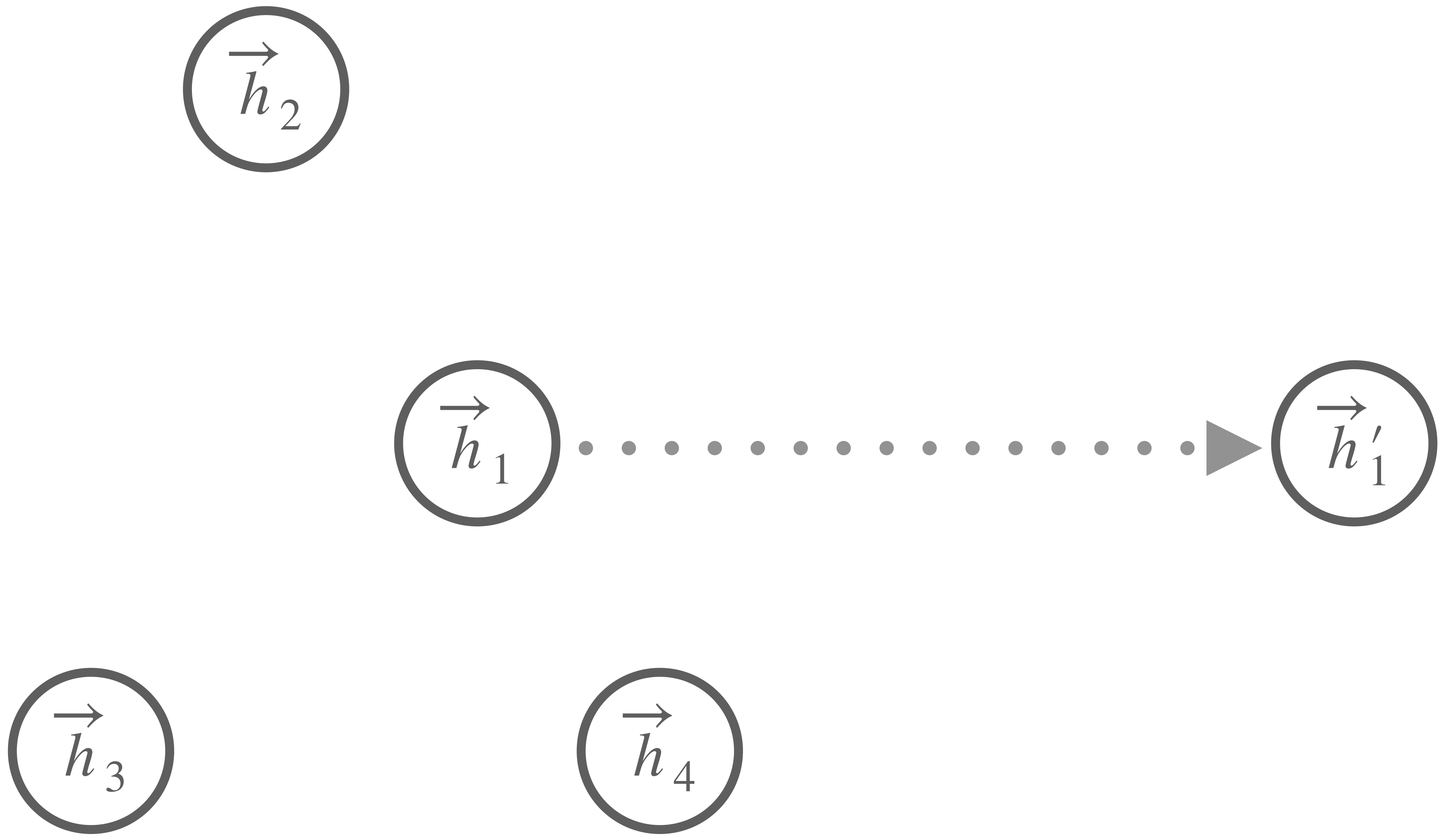
Linear transformation $\in \mathbb{R}^{F' \times F}$



Example of a GNN layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h'_i \in \mathbb{R}^{F'}$.

$$\vec{h}'_i = \sigma\left(\frac{1}{card(N_i)} \sum_{j \in N_i} \mathbf{W} \vec{h}_j \right)$$

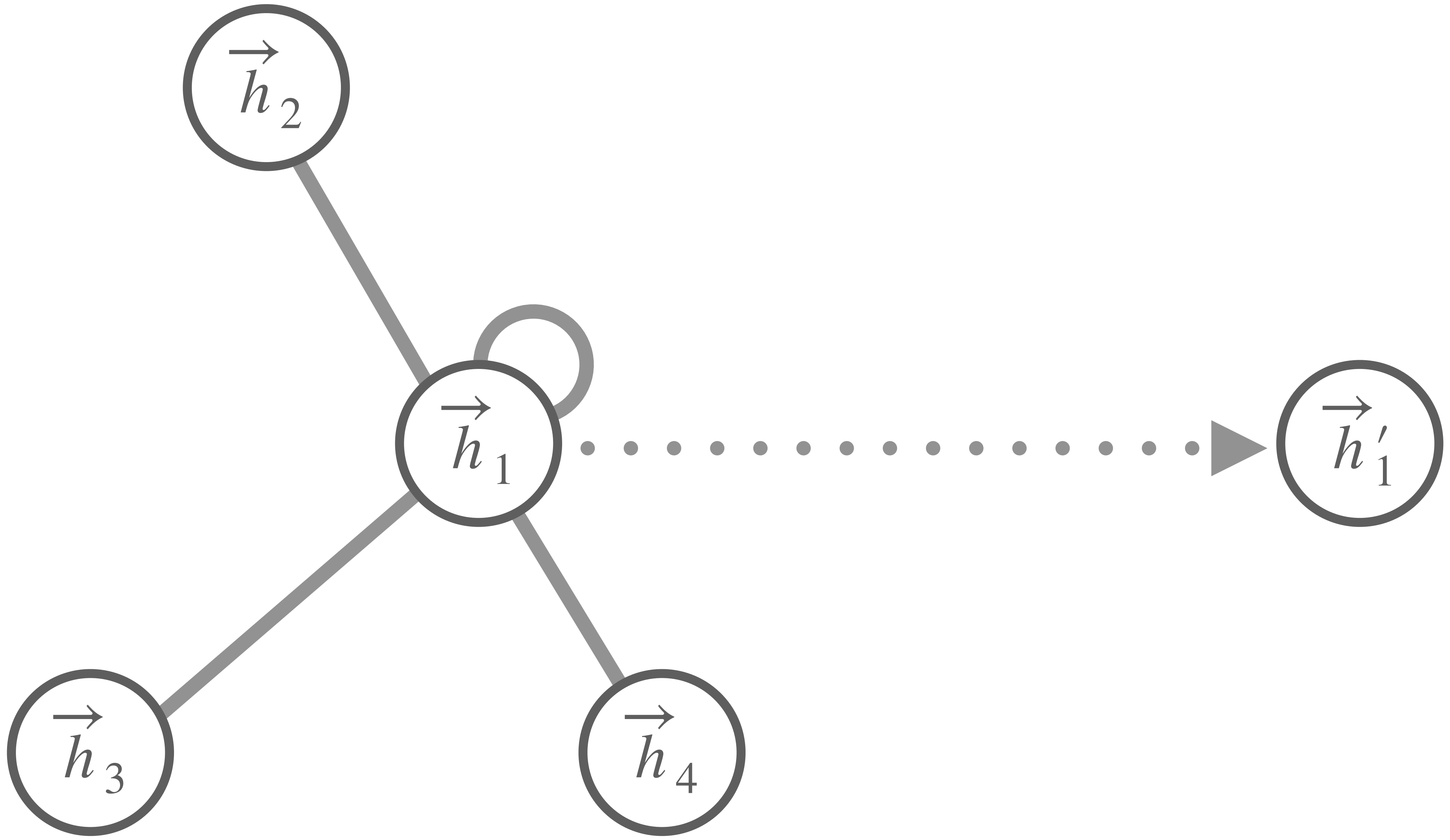


William L. Hamilton and Rex Ying and Jure Leskovec. 2018. Inductive Representation Learning on Large Graphs.

Example of a GNN layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h'_i \in \mathbb{R}^{F'}$.

$$\vec{h}'_i = \sigma\left(\frac{1}{card(N_i)} \sum_{j \in N_i} \mathbf{W} \vec{h}_j \right)$$

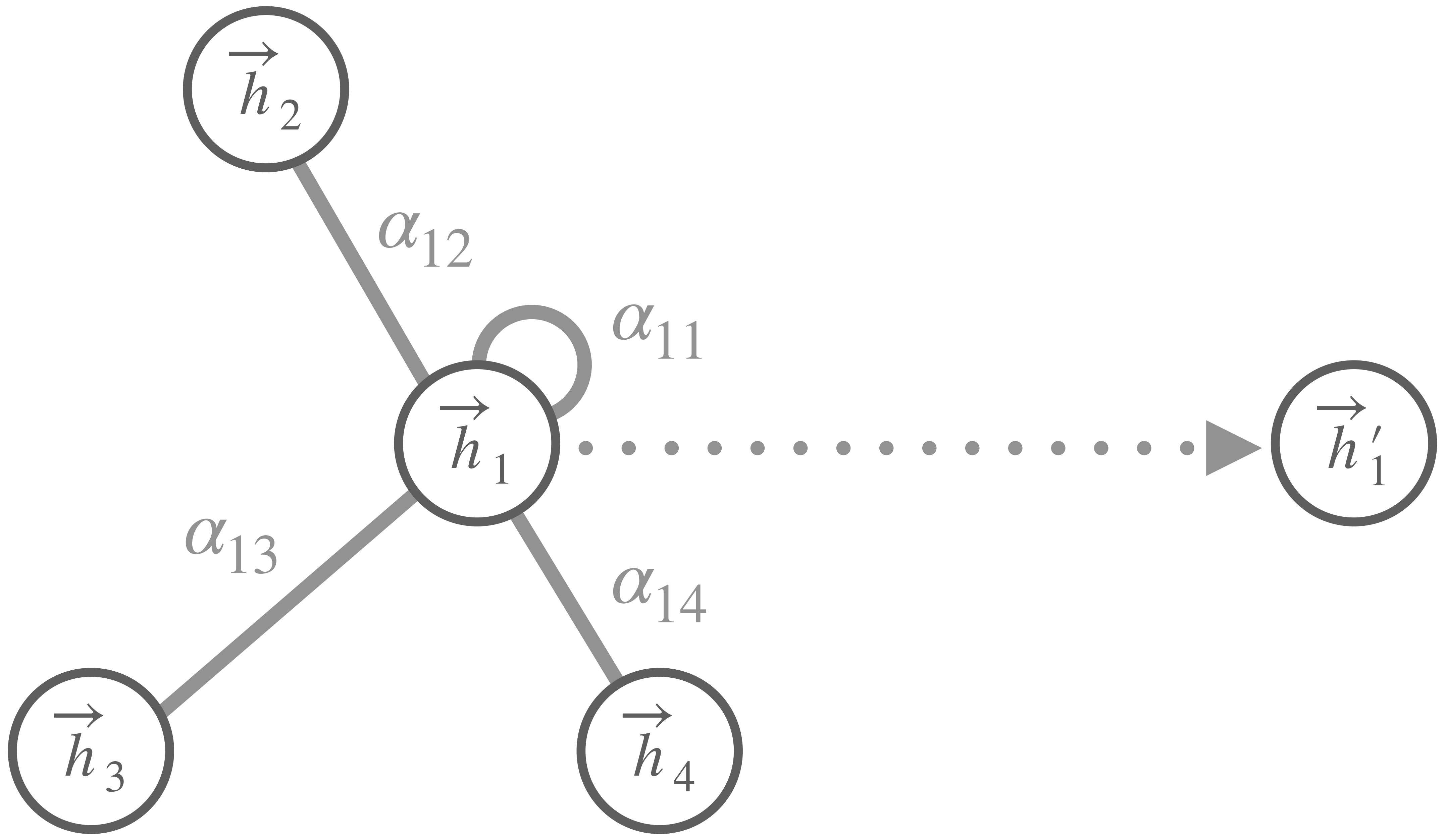


William L. Hamilton and Rex Ying and Jure Leskovec. 2018. Inductive Representation Learning on Large Graphs.

Graph Attention layer

Goal: Design a layer which takes nodes feature $h_i \in \mathbb{R}^F$ as input and output a new feature $h_i' \in \mathbb{R}^{F'}$.

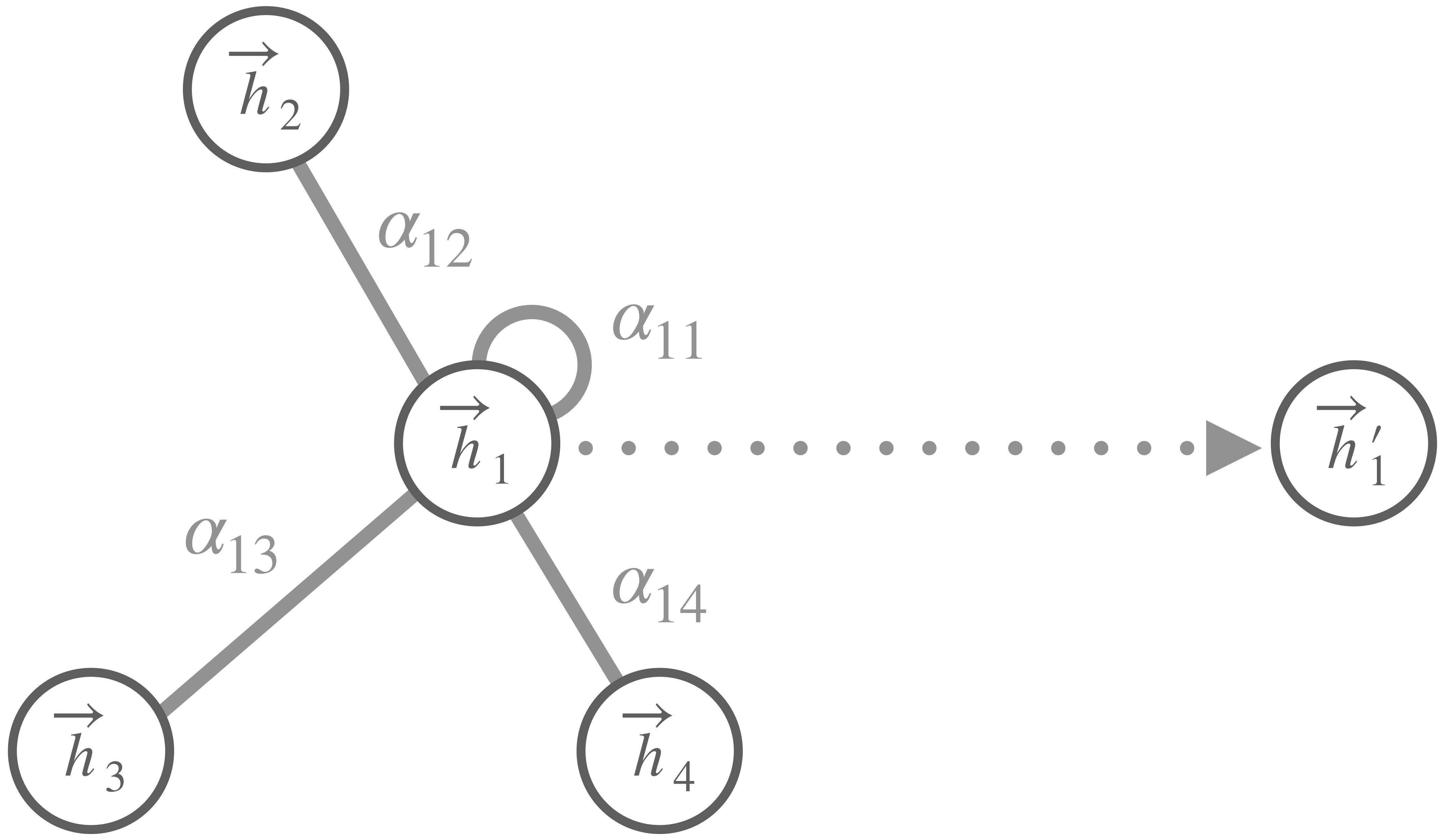
$$\vec{h}_i' = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \mathbf{W} \vec{h}_j \right)$$



Petar Veličković and Guillem Cucurull and Arantxa Casanova and Adriana Romero and Pietro Liò and Yoshua Bengio. 2018. Graph Attention Networks.

Attention on graphs?

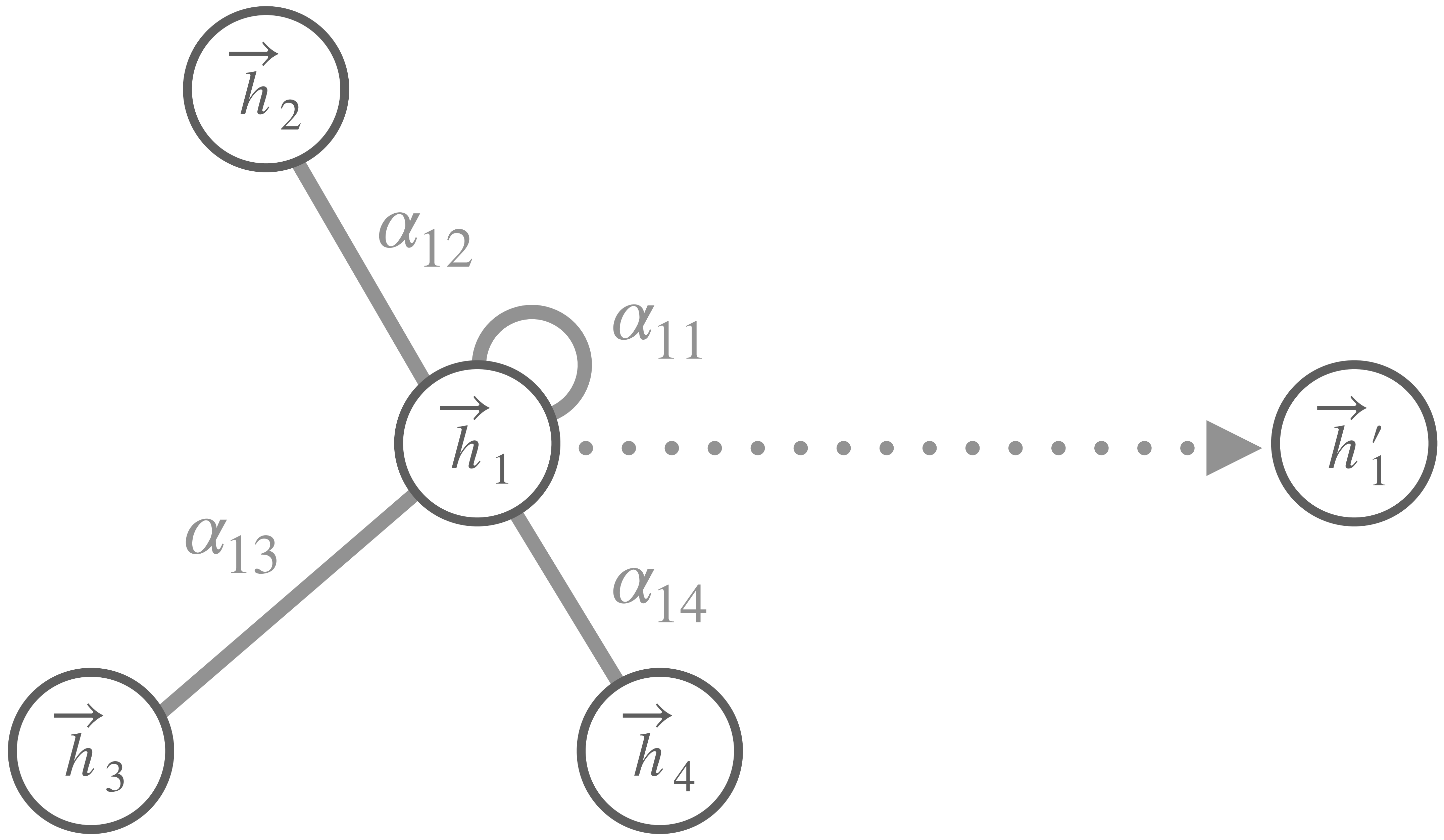
$$\vec{h}'_i = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \mathbf{W} \vec{h}_j\right)$$



Attention on graphs?

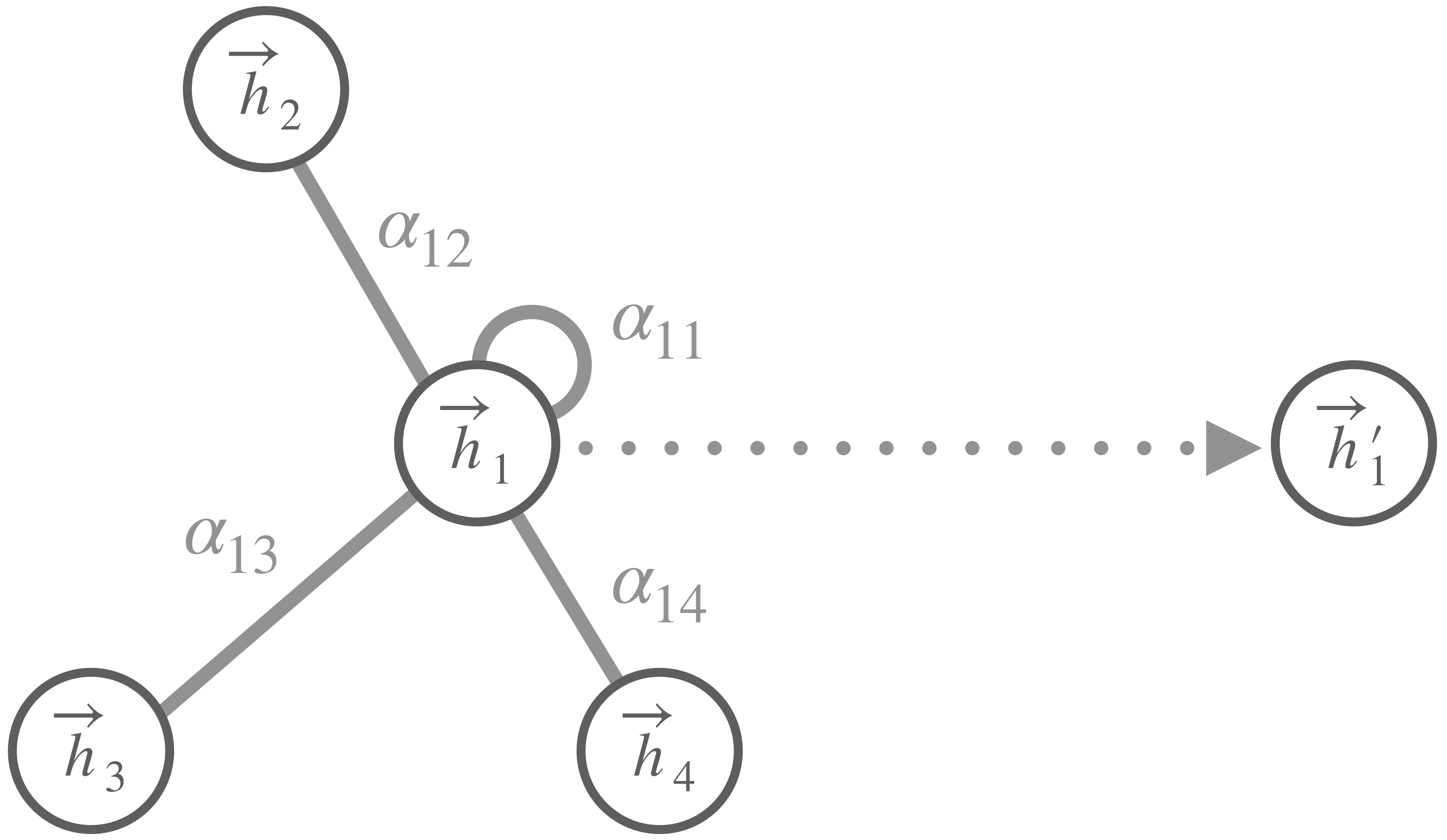
$$\vec{h}'_i = \sigma\left(\sum_{j \in N_i} \alpha_{ij}\, \mathbf{W} \vec{h}_j\right)$$

- α_{ij} represent the interest of $\vec{h}_j$ in the computation of $\vec{h}'_i$.



Attention on graphs?

$$\overrightarrow{h}'_i = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \, \mathbf{W} \, \overrightarrow{h}_j \right)$$



- α_{ij} represent the interest of $\overrightarrow{h}_j$ in the computation of $\overrightarrow{h}'_i$.

- coefficient easily comparable across different nodes.

Attentional mechanism

$$a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$$

Attentional mechanism

$$a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$$

$$e_{ij} = a(\mathbf{W}h_i, \mathbf{W}h_j)$$

Attentional mechanism

$$a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \rightarrow \mathbb{R}$$

$$e_{ij} = a(\mathbf{W}h_i, \mathbf{W}h_j)$$

If a is a single perceptron parametrized by a vector $\overrightarrow{a} \in \mathbb{R}^{2F'}$, and followed by a LeakyReLU activation function, the expression becomes:

Attentional mechanism

$$a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \to \mathbb{R}$$

$$e_{ij} = a(\mathbf{W}h_i, \mathbf{W}h_j)$$

If a is a single perceptron parametrized by a vector $\overrightarrow{a} \in \mathbb{R}^{2F'}$, and followed by a LeakyReLU activation function, the expression becomes:

$$e_{ij} = LeakyReLU\left(\overrightarrow{a}^T[\mathbf{W}h_i \,||\, \mathbf{W}h_j]\right)$$

$||$ designating the concatenation operator.

Attentional mechanism

$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathbf{N}_i} \exp(e_{ik})}$$

Attentional mechanism

$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in \mathbf{N}_i} \exp(e_{ik})}$$

$$= \frac{\exp\left(LeakyReLU\left(\overrightarrow{a}^T[\mathbf{W}h_i \,||\, \mathbf{W}h_j]\right)\right)}{\sum_{k \in \mathbf{N}_i} \exp\left(LeakyReLU\left(\overrightarrow{a}^T[\mathbf{W}h_i \,||\, \mathbf{W}h_k]\right)\right)}$$

Multi-head attention

Multi-head attention

$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \alpha_{ij}^{k} \, \mathbf{W}^{k} \vec{h}_j \right)$$

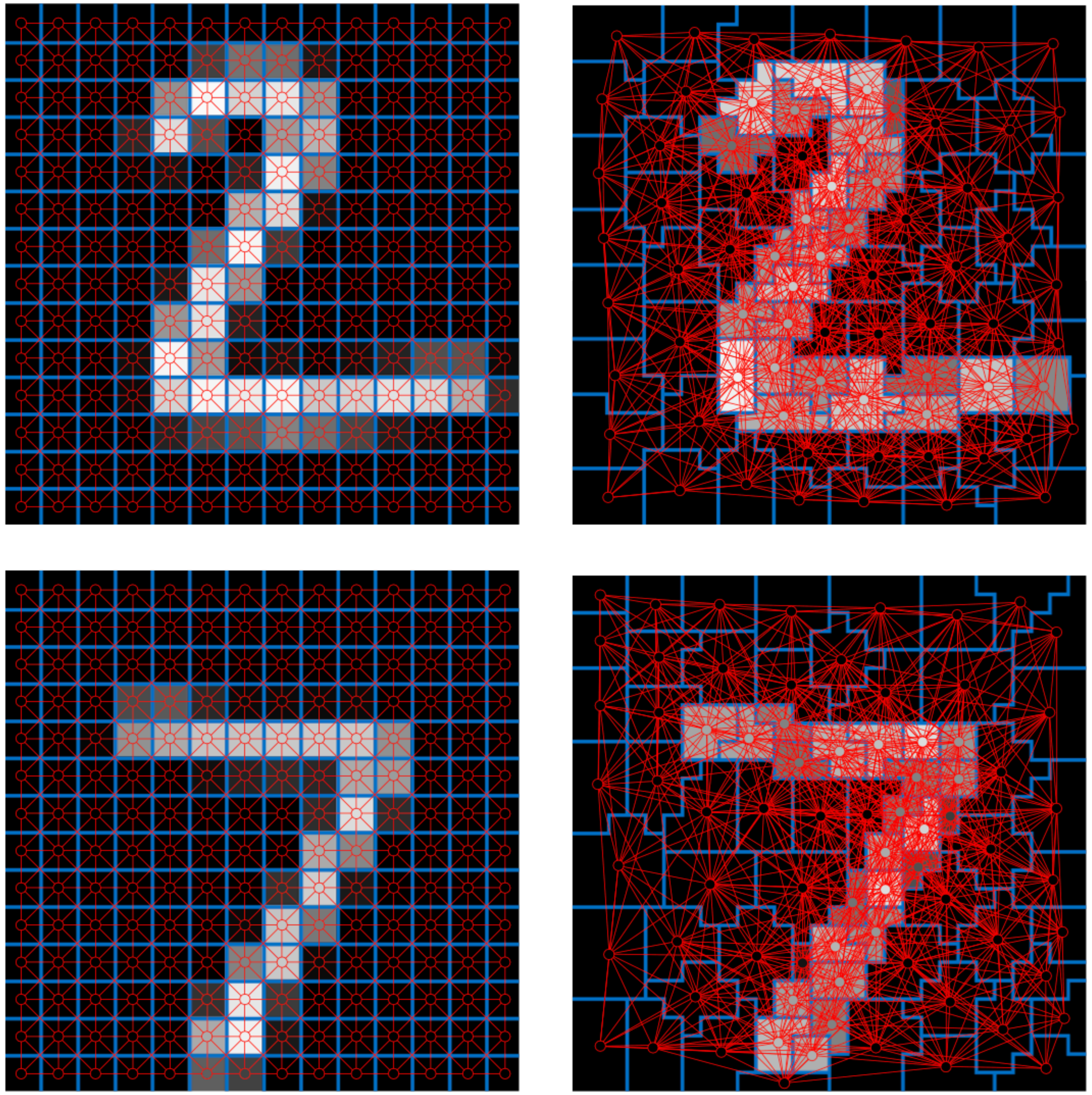
Multi-head attention

$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \alpha_{ij}^k \, \mathbf{W}^k \vec{h}_j \right)$$

Or, for the last layer:

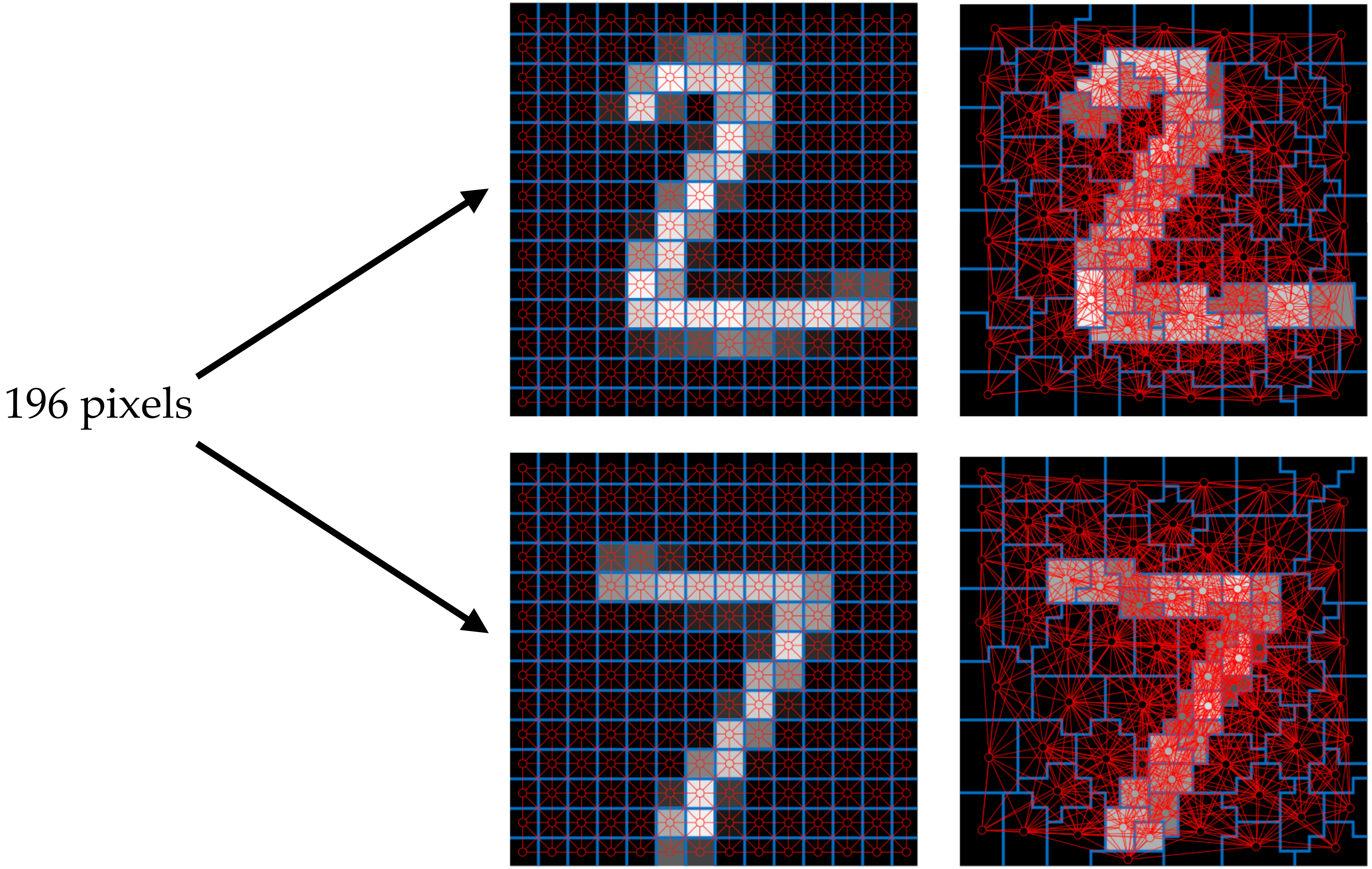
$$\vec{h}'_i = \sigma\left(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in N_i} \alpha_{ij}^k \, \mathbf{W}^k \vec{h}_j \right)$$

Example of application: superpixels



Regular grid Superpixels

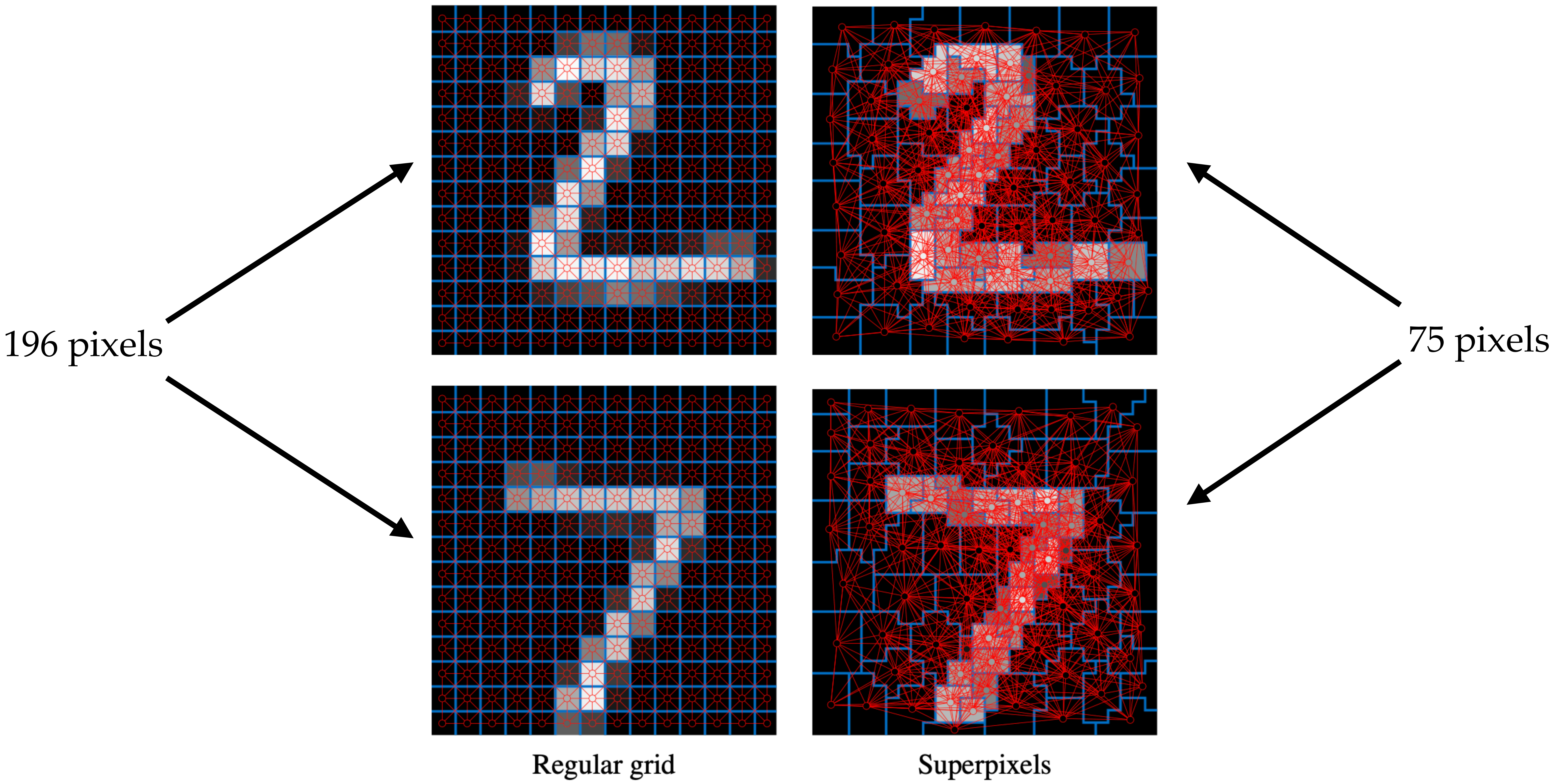
Example of application: superpixels



Regular grid

Superpixels

Example of application: superpixels

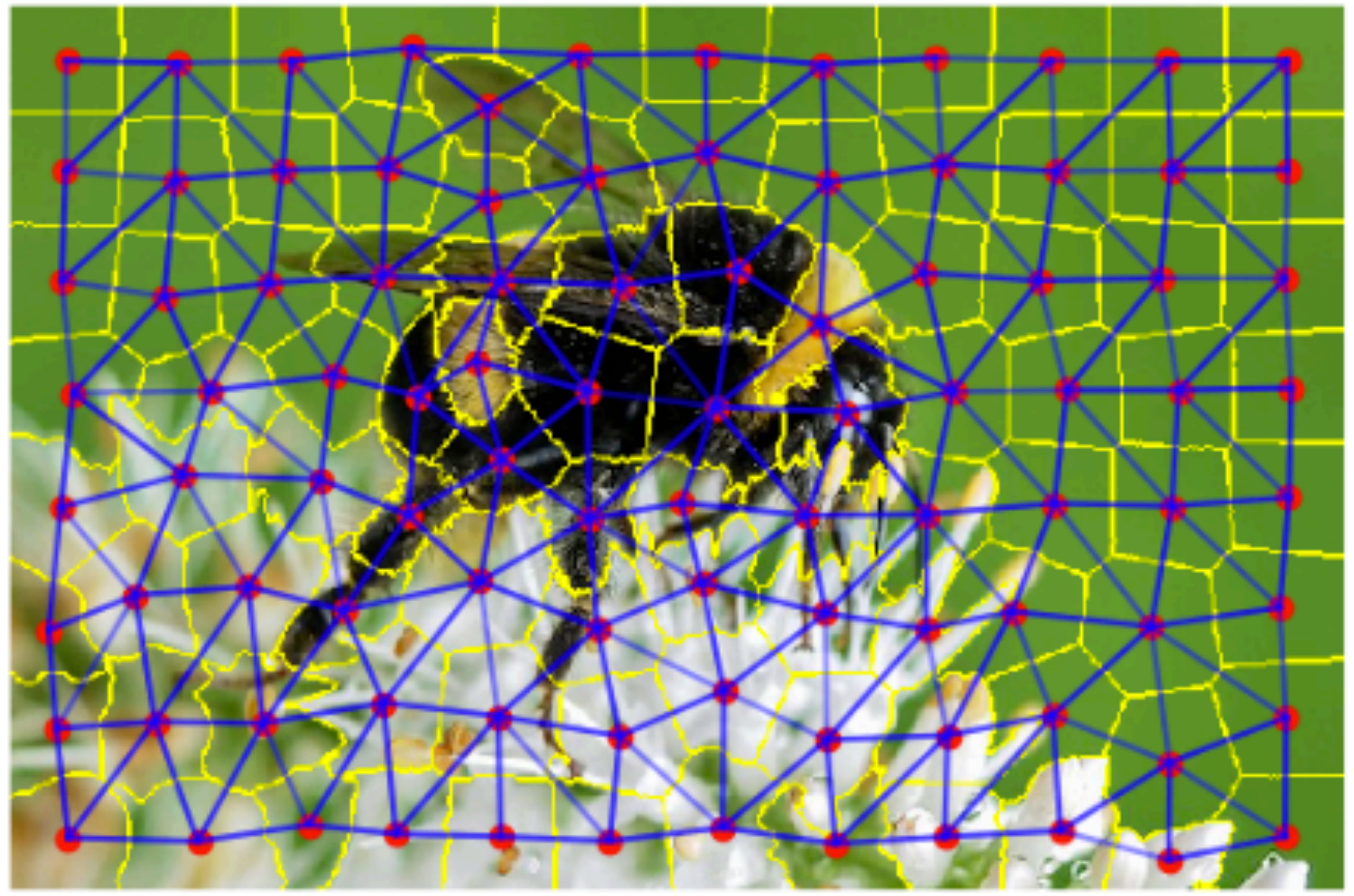


Example of application: superpixels

Input

📰 Pedro H. C. Avelar and Anderson R. Tavares and Thiago L. T. da Silveira and Cláudio R. Jung and Luís C. Lamb. 2020. Superpixel Image Classification with Graph Attention Networks.

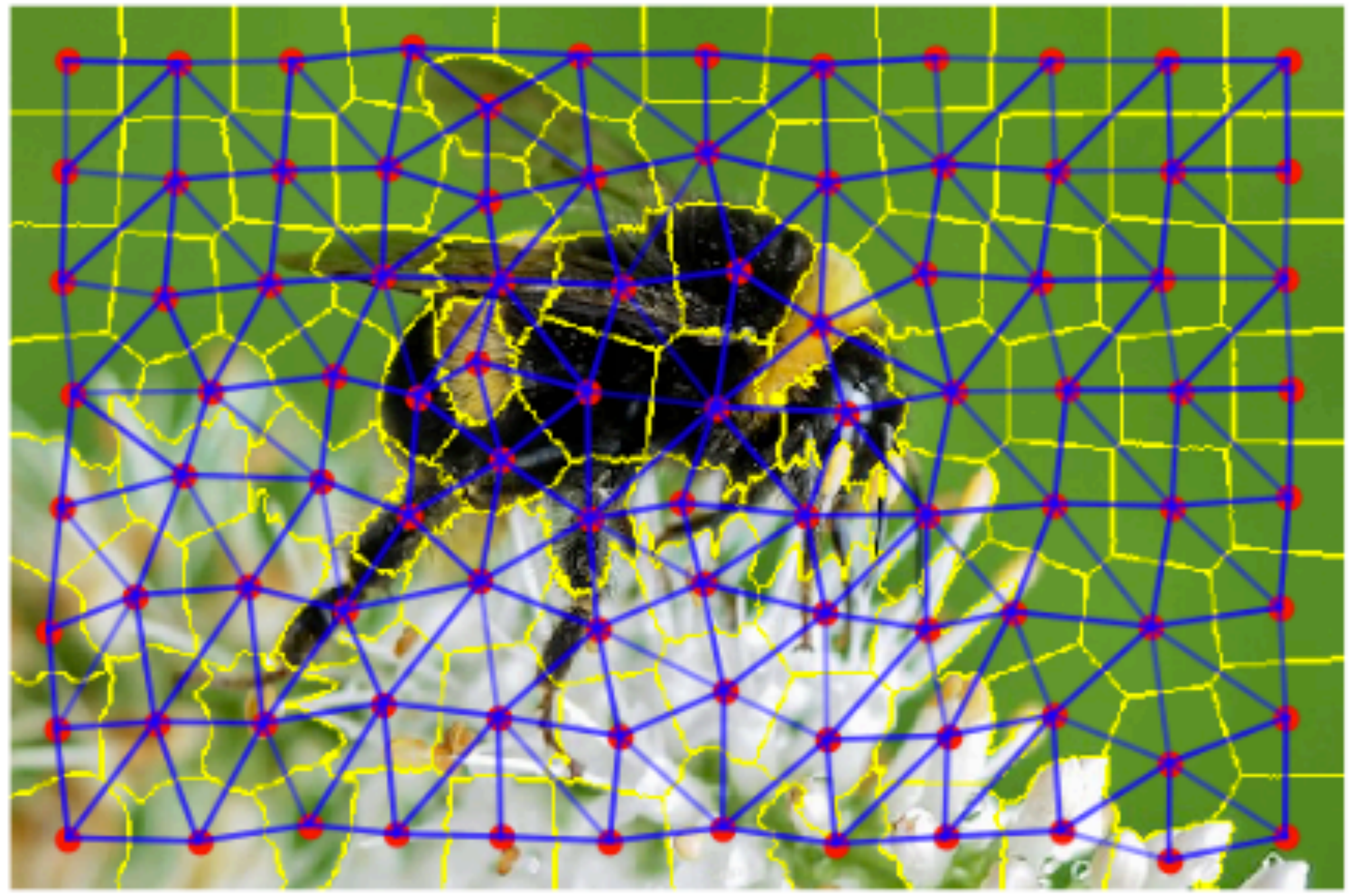
Example of application: superpixels



Input → SLIC

Pedro H. C. Avelar and Anderson R. Tavares and Thiago L. T. da Silveira and Cláudio R. Jung and Luís C. Lamb. 2020. Superpixel Image Classification with Graph Attention Networks.

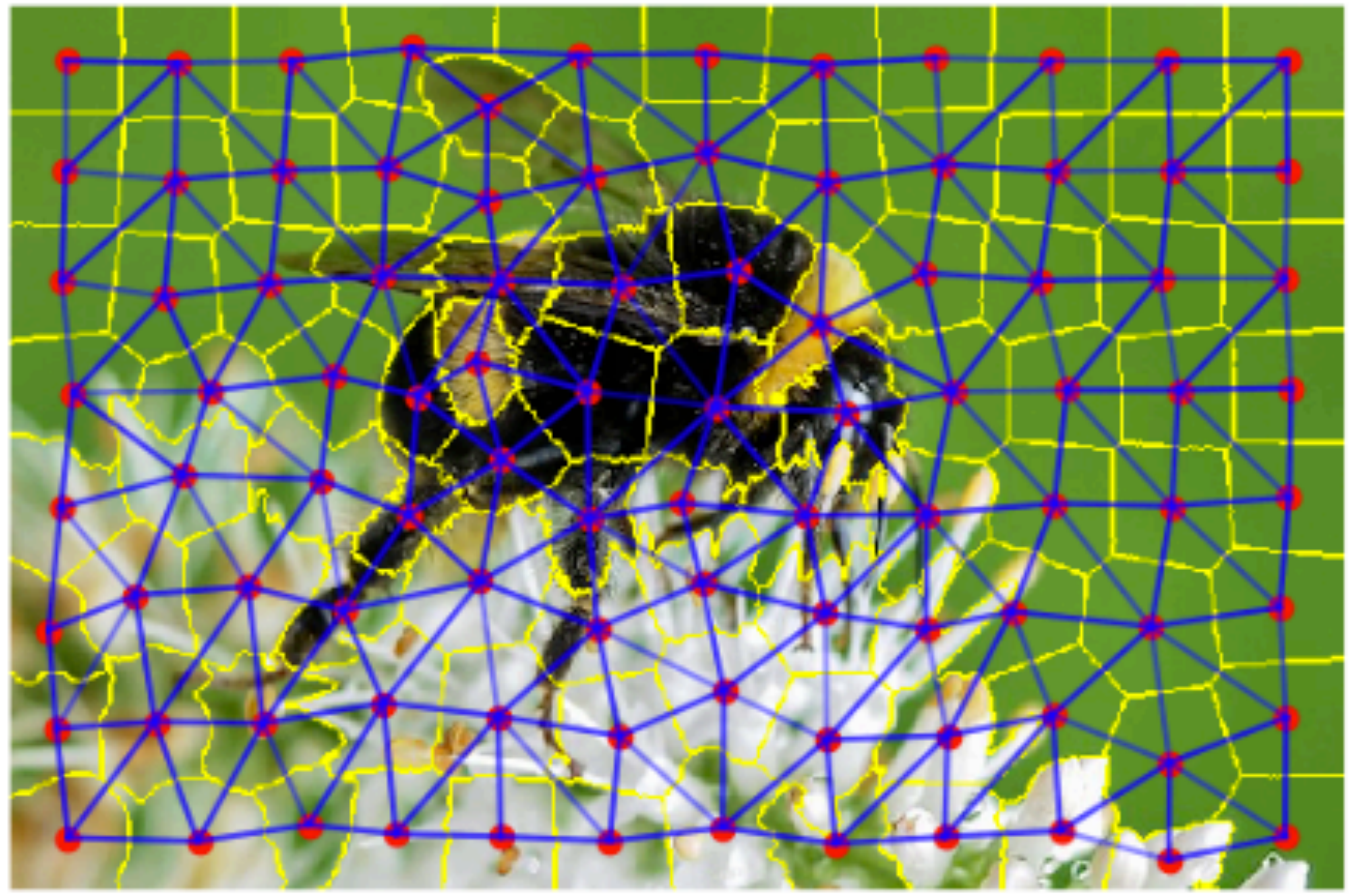
Example of application: superpixels



Input ⟶ SLIC ⟶ RAG

Pedro H. C. Avelar and Anderson R. Tavares and Thiago L. T. da Silveira and Cláudio R. Jung and Luís C. Lamb. 2020. Superpixel Image Classification with Graph Attention Networks.

Example of application: superpixels

Input → SLIC → RAG

Super pixel = (average luminosity, geometric centroid)

Pedro H. C. Avelar and Anderson R. Tavares and Thiago L. T. da Silveira and Cláudio R. Jung and Luís C. Lamb. 2020. Superpixel Image Classification with Graph Attention Networks.

Example of application: superpixels

MNISTs classification architecture

$\mathbb{R}^{3\times 75}$

Example of application: superpixels

MNISTs classification architecture

$$\xrightarrow{\mathbb{R}^{3\times 75}} \boxed{\text{GAL 1}} \xrightarrow{\mathbb{R}^{32\times 75}}$$

Example of application: superpixels

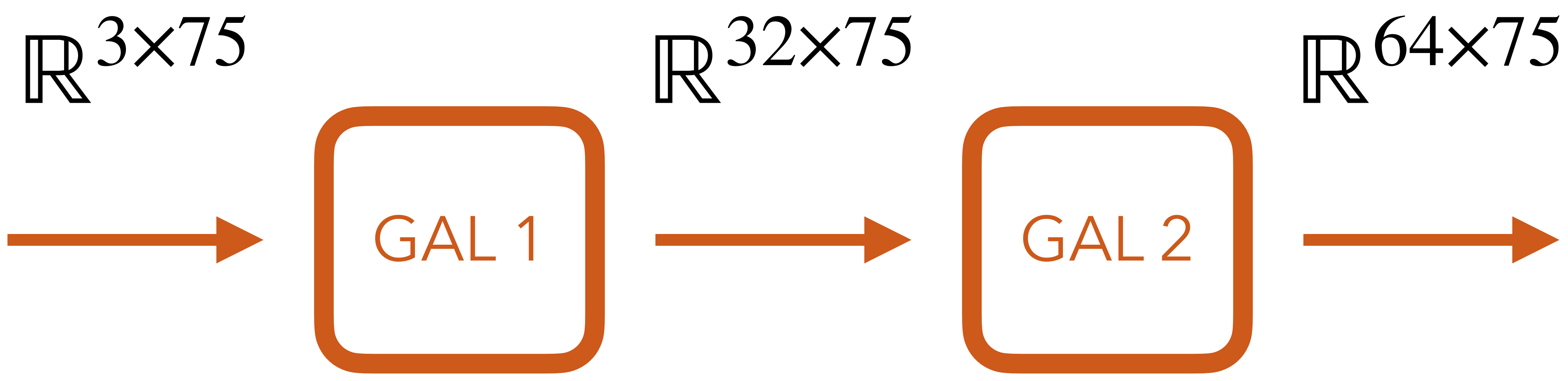
MNISTs classification architecture

$$\xrightarrow{\mathbb{R}^{3\times 75}} \boxed{\text{GAL 1}} \xrightarrow{\mathbb{R}^{32\times 75}}$$

GAL = Graph Attention Layer

Example of application: superpixels

MNISTs classification architecture



GAL = Graph Attention Layer

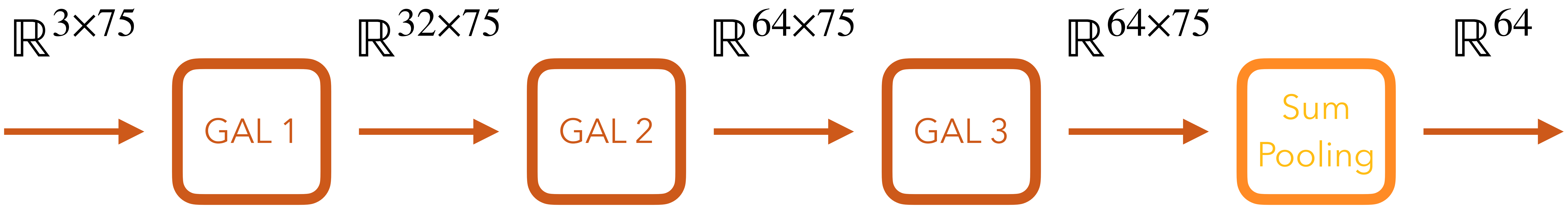
Example of application: superpixels

MNISTs classification architecture

GAL = Graph Attention Layer

Example of application: superpixels

MNISTs classification architecture



GAL = Graph Attention Layer

Example of application: superpixels

MNISTs classification architecture

GAL = Graph Attention Layer

Example of application: superpixels

	MNIST-75
MoNET [15]	91.11%
SplineCNN [6]	95.22%
GeoGCN [23]	95.95%
GAT-1Head	95.83%
GAT-2Head	**96.19**%


ATTENTION!!!
Thanks for your attention