Message Passing Attention Networks for Document Understanding

Michalis Vazirgiannis

Data Science and Mining Team (DaSciM), LIX
École Polytechnique, France and AUEB
http://www.lix.polytechnique.fr/dascim
Twitter: @mvazirg

June, 2020
Talk Outline

- Introduction to GNNs
- Message Passing GNNs
- Message Passing GNNs for Document Understanding
Traditional Node Representation

Representation: row of adjacency matrix
Traditional Node Representation

Representation: row of adjacency matrix

\[ \begin{pmatrix} 0 & 1 & \ldots & 0 \\ 1 & 0 & \ldots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \ldots & 0 \end{pmatrix} \]
Traditional Node Representation

Representation: row of adjacency matrix

However, such a representation suffers from:

- data sparsity
- high dimensionality

\[
\begin{pmatrix}
0 & 1 & \ldots & 0 \\
1 & 0 & \ldots & 1 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 1 & \ldots & 0 
\end{pmatrix}
\]
Node Embedding Methods

Map vertices of a graph into a low-dimensional space:

- dimensionality $d \ll |V|
- similar vertices are embedded close to each other in the low-dimensional space
Why Learning Node Representations?

Examples:
- Recommend friends
- Detect malicious users
Graph Classification

- Input data $G \in \mathcal{G}$
- Output $y \in \{-1, 1\}$
- Training set $S = \{(G_1, y_1), \ldots, (G_n, y_n)\}$
- Goal: estimate a function $f : \mathcal{G} \to \{-1, 1\}$ to predict $y$ from $f(G)$
Motivation - Protein Function Prediction

For each protein, create a graph that contains information about its

- structure
- sequence
- chemical properties

Perform graph classification to predict the function of proteins

[Borgwardt et al., Bioinformatics 2005]
Graph Regression

Input data $G \in \mathcal{G}$

Output $y \in \mathbb{R}$

Training set $S = \{(G_1, y_1), \ldots, (G_n, y_n)\}$

Goal: estimate a function $f : \mathcal{G} \rightarrow \mathbb{R}$ to predict $y$ from $f(G)$
Motivation - Molecular Property Prediction

12 targets corresponding to molecular properties: ['mu', 'alpha', 'HOMO', 'LUMO', 'gap', 'R2', 'ZPVE', 'U0', 'U', 'H', 'G', 'Cv']

Perform graph regression to predict the values of the properties

- SMILES: NC1=NCCC(=O)N1
  Targets: [2.54 64.1 -0.236 -2.79e-03 2.34e-01 900.7 0.12 -396.0 -396.0 -396.0 -396.0 26.9]

- SMILES: CN1CCC(=O)C1=N
  Targets: [4.218 68.69 -0.224 -0.056 0.168 914.65 0.131 -379.959 -379.951 -379.95 -379.992 27.934]

- SMILES: N=C1OC2CC1C(=O)O2
  Targets: [4.274 61.94 -0.282 -0.026 0.256 887.402 0.104 -473.876 -473.87 -473.869 -473.907 24.823]

- SMILES: C1N2C3C4C5OC13C2C5

Perform graph regression to predict the values of the properties

- DFT
  Targets: \( E, \omega_0, \ldots \)

- Message Passing Neural Net
  \( \sim 10^{-2} \) seconds

- \([\text{Gilmer et al., ICML'17}]\)
Message Passing Neural Networks

Idea: Each node exchanges messages with its neighbors and updates its representations based on these messages.

The message passing scheme runs for $T$ time steps and updates the representation of each vertex $h^t_v$ based on its previous representation and the representations of its neighbors:

$$m^{t+1}_v = \sum_{u \in \mathcal{N}(v)} M_t(h^t_v, h^t_u, e_{vu})$$

$$h^{t+1}_v = U_t(h^t_v, m^{t+1}_v)$$

where $\mathcal{N}(v)$ is the set of neighbors of $v$ and $M_t, U_t$ are message functions and vertex update functions respectively.
Example of Message Passing Scheme

\[ h_{1}^{t+1} = W_0 h_1^t + W_1 h_2^t + W_1 h_3^t \]
\[ h_{2}^{t+1} = W_0 h_2^t + W_1 h_1^t + W_1 h_2^t + W_1 h_4^t \]
\[ h_{3}^{t+1} = W_0 h_3^t + W_1 h_1^t + W_1 h_2^t + W_1 h_4^t \]
\[ h_{4}^{t+1} = W_0 h_4^t + W_1 h_2^t + W_1 h_3^t + W_1 h_5^t \]
\[ h_{5}^{t+1} = W_0 h_5^t + W_1 h_4^t + W_1 h_6^t \]
\[ h_{6}^{t+1} = W_0 h_6^t + W_1 h_5^t \]

Remark: Biases are omitted for clarity
Output of message passing phase:

\[ \{ h_1^{T_{max}}, h_2^{T_{max}}, h_3^{T_{max}}, h_4^{T_{max}}, h_5^{T_{max}}, h_6^{T_{max}} \} \]

Graph representation:

\[
\mathbf{z}_G = \frac{1}{6} ( h_1^{T_{max}} + h_2^{T_{max}} + h_3^{T_{max}} + h_4^{T_{max}} + h_5^{T_{max}} + h_6^{T_{max}} )
\]
Message Passing using Matrix Multiplication

- Let $v_1$ denote some node and $N(v_1) = \{v_2, v_3\}$ where $N(v_1)$ is the set of neighbors of $v_1$

- A common update scheme is:
  \[ h_{1}^{t+1} = W^t h_{1}^t + W^t h_{2}^t + W^t h_{3}^t \]

- The above update scheme can be rewritten as:
  \[ h_{1}^{t+1} = \sum_{i \in N(v_1) \cup \{v_1\}} W^t h_{i}^t \]

- In matrix form (for all the nodes), this is equivalent to:
  \[ H^{t+1} = (A + I) H^t W^t \]

where $A$ is the adjacency matrix of the graph, $I$ the identity matrix, and $H^t$ a matrix that contains the node representations at time step $t$ (as rows)
GCN

- Utilizes a variant of the above message passing scheme
- Given the adjacency matrix $A$ of a graph, GCN first computes the following normalized matrix:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$$

where

$\tilde{A} = A + I$

$\tilde{D}$: a diagonal matrix such that $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$

- Normalization helps to avoid numerical instabilities and exploding/vanishing gradients

- Then, the output of the model is:

$$Z = \text{softmax}(\hat{A} \ \text{ReLU}(\hat{A} \ X \ W^0) \ W^1)$$

where

$X$: contains the attributes of the nodes, i.e., $H^0$

$W^0, W^1$: trainable weight matrices for $t = 0$ and $t = 1$

[Kipf and Welling, ICLR’17]
To learn node embeddings, GCN minimizes the following loss function:

\[ \mathcal{L} = - \sum_{i \in I} \sum_{j=1}^{\left| \mathcal{C} \right|} Y_{ij} \log \hat{Y}_{ij} \]

\( I \): indices of the nodes of the training set
\( \mathcal{C} \): set of class labels
Experimental Evaluation

Experimental comparison conducted in [Kipf and Welling, ICLR’17]

Compared algorithms:

- DeepWalk
- ICA [2]
- Planetoid
- GCN

Task: node classification
Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>Nodes</th>
<th>Edges</th>
<th>Classes</th>
<th>Features</th>
<th>Label rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citeseer</td>
<td>Citation network</td>
<td>3,327</td>
<td>4,732</td>
<td>6</td>
<td>3,703</td>
<td>0.036</td>
</tr>
<tr>
<td>Cora</td>
<td>Citation network</td>
<td>2,708</td>
<td>5,429</td>
<td>7</td>
<td>1,433</td>
<td>0.052</td>
</tr>
<tr>
<td>Pubmed</td>
<td>Citation network</td>
<td>19,717</td>
<td>44,338</td>
<td>3</td>
<td>500</td>
<td>0.003</td>
</tr>
<tr>
<td>NELL</td>
<td>Knowledge graph</td>
<td>65,755</td>
<td>266,144</td>
<td>210</td>
<td>5,414</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Label rate: number of labeled nodes that are used for training divided by the total number of nodes

Citation network datasets:

- nodes are documents and edges are citation links
- each node has an attribute (the bag-of-words representation of its abstract)

NELL is a bipartite graph dataset extracted from a knowledge graph
### Classification accuracies of the 4 methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Citeseer</th>
<th>Cora</th>
<th>Pubmed</th>
<th>NELL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepWalk</td>
<td>43.2</td>
<td>67.2</td>
<td>65.3</td>
<td>58.1</td>
</tr>
<tr>
<td>ICA</td>
<td>69.1</td>
<td>75.1</td>
<td>73.9</td>
<td>23.1</td>
</tr>
<tr>
<td>Planetoid</td>
<td>64.7 (26s)</td>
<td>75.7 (13s)</td>
<td>77.2 (25s)</td>
<td>61.9 (185s)</td>
</tr>
<tr>
<td>GCN</td>
<td><strong>70.3</strong> (7s)</td>
<td><strong>81.5</strong> (4s)</td>
<td><strong>79.0</strong> (38s)</td>
<td><strong>66.0</strong> (48s)</td>
</tr>
</tbody>
</table>

**Observation:** DeepWalk → unsupervised learning of embeddings

→ fails to compete against the supervised approaches
**Goal:** Apply the Message Passing (MP) framework to representation learning on text
→ documents/sentences represented as word co-occurrence networks

**Related work:**
The MP framework has been applied to graph representations of text where nodes represent:

- documents → edge weights equal to distance between BoW representations of documents [Henaff et al., arXiv’15]
- documents and terms → document-term edges are weighted by TF-IDF and term-term edges by pointwise mutual information [Yao et al., AAAI’19]
- terms → all document graphs have identical structure, but different node attributes (based on some term weighting scheme). Each term connected to its $k$ most similar terms [Defferrard et al., NIPS’16]
Word Co-occurrence Networks

Each document is represented as a graph $G = (V, E)$ consisting of a set $V$ of vertices and a set $E$ of edges between them.

- Vertices → unique terms
- Edges → co-occurrences within a fixed-size sliding window
- Vertex attributes → embeddings of terms

Graph representation more flexible than $n$-grams

Figure: Graph representation of doc: “to be or not to be: that is the question”. [Rousseau and Vazirgiannis, CIKM’13]
Message Passing Neural Networks

Use Message Passing Neural Networks (MPNNs) to perform text categorization consist of two steps:

**Step 1:** At time $t + 1$, a message vector $m_{v}^{t+1}$ is computed from the representations of the neighbors $\mathcal{N}(v)$ of $v$:

$$m_{v}^{t+1} = \text{AGGREGATE}^{t+1}\left(\{h_{w}^{t} \mid w \in \mathcal{N}(v)\}\right)$$

The new representation $h_{v}^{t+1}$ of $v$ is then computed by combining its current feature vector $h_{v}^{t}$ with the message vector $m_{v}^{t+1}$:

$$h_{v}^{t+1} = \text{COMBINE}^{t+1}(h_{v}^{t}, m_{v}^{t+1})$$

Messages are passed for $T$ time steps

**Step 2:** To produce a graph-level feature vector, a READOUT pooling function, that must be invariant to permutations, is applied:

$$h_{G} = \text{READOUT}\left(\{h_{v}^{T} \mid v \in \mathcal{V}\}\right)$$
Message Passing Attention Networks for Document Understanding (MPAD)

- Represent textual documents as word co-occurrence networks
  transform text mining problems into graph mining problems
- Employ graph neural networks (e.g., MPNNs) to deal with machine learning problems in text mining
  - text categorization
  - question answering
  - text embedding
- MPAD belongs to the family of MPNNs
  - nodes (i.e. words) update their representations by exchanging messages with their neighbors (i.e. words in their context)
  - a self-attention mechanism is employed to produce document/sentence (i.e. graph) representations from node (i.e. word) representations
**Master node**: Generated networks also contain a special document node, linked to all other nodes
- can encode a summary of the document

**Figure**: Graph representation of the document: “to be or not to be: that is the question”. The black node corresponds to the master node
MPAD utilizes the following AGGREGATE function:

\[
X^{t+1} = MLP^{t+1}(H^t) \\
M^{t+1} = D^{-1} A X^{t+1}
\]  

- **A** ⇒ adjacency matrix of word co-occurrence network
- **D** ⇒ a diagonal matrix such that \(D_{ii} = \sum_j A_{ij}\)
- **\(H^t \in \mathbb{R}^{n \times d}\)** ⇒ contains node features (**\(H^0\)** contains word (node) embeddings)
- Renormalization ⇒ matrix product \(D^{-1} A X^{t+1}\) computes average of neighbors’ features
  ↦ avoids numerical instabilities
Step 1: Message Passing (2/2)

- The COMBINE function corresponds to a GRU:

  $$
  H^{t+1} = GRU(H^t, M^{t+1})
  $$

  $$
  R^{t+1} = \sigma(W_R^{t+1}M^{t+1} + U_R^{t+1}X^{t+1})
  $$

  $$
  Z^{t+1} = \sigma(W_Z^{t+1}M^{t+1} + U_Z^{t+1}X^{t+1})
  $$

  $$
  \tilde{H}^{t+1} = \tanh(W^{t+1}M^{t+1} + U^{t+1}(R^{t+1} \odot X^{t+1}))
  $$

  $$
  H^{t+1} = (1 - Z^{t+1}) \odot X^{t+1} + Z^{t+1} \odot \tilde{H}^{t+1}
  $$

where $W, U$ are trainable weight matrices

- $R \Rightarrow$ reset gate controls amount of information from the previous time step that should propagate to the candidate representations $\tilde{H}^{t+1}$:
- $Z \Rightarrow$ update gate

After performing updates for $T$ iterations, we obtain a matrix $H^T \in \mathbb{R}^{n \times d}$ containing the final vertex representations.
Step 2: Readout

Let $\hat{H}^T \in \mathbb{R}^{(n-1) \times d}$ be the representation matrix without the row of the master node. The READOUT function applies self-attention to $\hat{H}^T$:

$$Y^T = \tanh(\hat{H}^T W_A^T)$$

$$\alpha_i^T = \frac{\exp(Y_i^T \cdot v^T)}{\sum_{j=1}^{n-1} \exp(Y_j^T \cdot v^T)}$$

$$u^T = \sum_{i=1}^{n-1} \alpha_i^T \hat{H}_i^T$$

Then, $u^T$ is concatenated with the master node representation

**Multi-readout**: apply readout to all time steps and concatenate the results, finally obtaining $h_G \in \mathbb{R}^{T \times 2d}$:

$$h_G = \text{CONCAT}(\text{READOUT}(H^t) \mid t = 1 \ldots T)$$
Hierarchical Variants

MPAD applied to *sentences* instead of documents $\rightarrow$ sentence representations $\leftarrow$ sentence representations combined to produce document representations:

**MPAD-sentence-att:**
- sentence embeddings are combined through self-attention

**MPAD-clique/path:**
- documents modeled as graphs where nodes represent sentences
- two types of graphs:
  - **MPAD-clique:** complete graphs
  - **MPAD-path:** path graphs where two nodes are linked by a directed edge if the two sentences follow each other in the document
- graph is then fed to a different MPAD instance (no *master node*)
- feature vectors of nodes initialized with sentence embeddings previously obtained

**Figure:** Illustration of MPAD-path (☉: master node).
Experimental Evaluation

Task: Text Categorization

Datasets: 10 standard benchmark datasets, covering the topic identification, coarse and fine sentiment analysis and opinion mining, and subjectivity detection tasks

<table>
<thead>
<tr>
<th>Dataset</th>
<th># training examples</th>
<th># test examples</th>
<th># classes</th>
<th>av. # words</th>
<th>max # words</th>
<th>voc. size</th>
<th># pretrained words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reuters</td>
<td>5,485</td>
<td>2,189</td>
<td>8</td>
<td>102.3</td>
<td>964</td>
<td>23,585</td>
<td>15,587</td>
</tr>
<tr>
<td>BBCSport</td>
<td>737</td>
<td>CV</td>
<td>5</td>
<td>380.5</td>
<td>1,818</td>
<td>14,340</td>
<td>13,390</td>
</tr>
<tr>
<td>Polarity</td>
<td>10,662</td>
<td>CV</td>
<td>2</td>
<td>20.3</td>
<td>56</td>
<td>18,777</td>
<td>16,416</td>
</tr>
<tr>
<td>Subjectivity</td>
<td>10,000</td>
<td>CV</td>
<td>2</td>
<td>23.3</td>
<td>120</td>
<td>21,335</td>
<td>17,896</td>
</tr>
<tr>
<td>MPQA</td>
<td>10,606</td>
<td>CV</td>
<td>2</td>
<td>3.0</td>
<td>36</td>
<td>6,248</td>
<td>6,085</td>
</tr>
<tr>
<td>IMDB</td>
<td>25,000</td>
<td>25,000</td>
<td>2</td>
<td>254.3</td>
<td>2,633</td>
<td>141,655</td>
<td>104,391</td>
</tr>
<tr>
<td>TREC</td>
<td>5,452</td>
<td>500</td>
<td>6</td>
<td>10.0</td>
<td>37</td>
<td>9,593</td>
<td>9,125</td>
</tr>
<tr>
<td>SST-1</td>
<td>157,918</td>
<td>2,210</td>
<td>5</td>
<td>7.4</td>
<td>53</td>
<td>17,833</td>
<td>16,262</td>
</tr>
<tr>
<td>SST-2</td>
<td>77,833</td>
<td>1,821</td>
<td>2</td>
<td>9.5</td>
<td>53</td>
<td>17,237</td>
<td>15,756</td>
</tr>
<tr>
<td>Yelp2013</td>
<td>301,514</td>
<td>33,504</td>
<td>5</td>
<td>143.7</td>
<td>1,184</td>
<td>48,212</td>
<td>48,212</td>
</tr>
</tbody>
</table>

Table: Statistics of the datasets used in our experiments. CV indicates that cross-validation was used. # pretrained words refers to the number of words in the vocabulary having an entry in the Google News word vectors (except for Yelp2013).
### Text Categorization Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Reut.</th>
<th>BBC</th>
<th>Pol.</th>
<th>Subj.</th>
<th>MPQA</th>
<th>IMDB</th>
<th>TREC</th>
<th>SST-1</th>
<th>SST-2</th>
<th>Yelp’13</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc2vec</td>
<td>95.34</td>
<td>98.64</td>
<td>67.30</td>
<td>88.27</td>
<td>82.57</td>
<td>92.5</td>
<td>70.80</td>
<td>48.7</td>
<td>87.8</td>
<td>57.7</td>
</tr>
<tr>
<td>CNN</td>
<td>97.21</td>
<td>98.37</td>
<td>81.5</td>
<td>93.4</td>
<td>89.5</td>
<td>90.28</td>
<td>93.6</td>
<td>48.0</td>
<td>87.2</td>
<td>64.89</td>
</tr>
<tr>
<td>DAN</td>
<td>94.79</td>
<td>94.30</td>
<td>80.3</td>
<td>92.44</td>
<td>88.91</td>
<td>89.4</td>
<td>89.60</td>
<td>47.7</td>
<td>86.3</td>
<td>61.55</td>
</tr>
<tr>
<td>Tree-LSTM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>51.0</td>
<td>88.0</td>
<td>-</td>
</tr>
<tr>
<td>DRNN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>49.8</td>
<td>86.6</td>
<td>-</td>
</tr>
<tr>
<td>LSTMN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>47.9</td>
<td>87.0</td>
<td>-</td>
</tr>
<tr>
<td>C-LSTM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>94.6</td>
<td>49.2</td>
<td>87.8</td>
</tr>
<tr>
<td>SPGK</td>
<td>96.39</td>
<td>94.97</td>
<td>77.89</td>
<td>91.48</td>
<td>85.78</td>
<td>00M</td>
<td>90.69</td>
<td>00M</td>
<td>00M</td>
<td>00M</td>
</tr>
<tr>
<td>WMD</td>
<td>96.5</td>
<td>98.71</td>
<td>66.42</td>
<td>86.04</td>
<td>83.95</td>
<td>00M</td>
<td>73.40</td>
<td>00M</td>
<td>00M</td>
<td>00M</td>
</tr>
<tr>
<td>DiSAN</td>
<td>97.35</td>
<td>96.05</td>
<td>80.38</td>
<td>94.2</td>
<td>90.1</td>
<td>83.25</td>
<td>94.2</td>
<td>51.72</td>
<td>86.76</td>
<td>60.51</td>
</tr>
<tr>
<td>LSTM-GRNN</td>
<td>96.16</td>
<td>95.52</td>
<td>79.98</td>
<td>92.38</td>
<td>89.08</td>
<td>89.98</td>
<td>89.40</td>
<td>48.09</td>
<td>86.38</td>
<td>65.1</td>
</tr>
<tr>
<td>HN-ATT</td>
<td>97.25</td>
<td>96.73</td>
<td>80.78</td>
<td>92.92</td>
<td>89.08</td>
<td>90.06</td>
<td>90.80</td>
<td>49.00</td>
<td>86.71</td>
<td>68.2</td>
</tr>
<tr>
<td>MPAD</td>
<td>97.07</td>
<td>98.37</td>
<td>80.24</td>
<td>93.46*</td>
<td>90.02</td>
<td>91.30</td>
<td>95.60*</td>
<td>49.09</td>
<td>87.80</td>
<td>66.16</td>
</tr>
<tr>
<td>MPAD-sentence-att</td>
<td>96.89</td>
<td>99.32</td>
<td>80.44</td>
<td>93.02</td>
<td>90.12*</td>
<td>91.70</td>
<td>95.60*</td>
<td>49.95*</td>
<td>88.30*</td>
<td>66.47</td>
</tr>
<tr>
<td>MPAD-clique</td>
<td><strong>97.57</strong></td>
<td><strong>99.72</strong></td>
<td>81.17*</td>
<td>92.82</td>
<td>89.96</td>
<td>91.87*</td>
<td>95.20</td>
<td>48.86</td>
<td>87.91</td>
<td>66.60</td>
</tr>
<tr>
<td>MPAD-path</td>
<td>97.44</td>
<td>99.59</td>
<td>80.46</td>
<td>93.31</td>
<td>89.81</td>
<td>91.84</td>
<td>93.80</td>
<td>49.68</td>
<td>87.75</td>
<td>66.80*</td>
</tr>
</tbody>
</table>

**Table:** Classification accuracies. Best performance per column in **bold**, *best MPAD variant. 00M: >16GB RAM.
Ablation Study

- undirected: undirected word co-occurrence networks
- no master node: word co-occurrence networks without master nodes
- no renormalization: do not multiply by $D^{-1}$ in Eq. (1)
- neighbors-only: do not use COMBINE function (Eq. (2)) and set $H^{t+1} = M^{t+1}$
- no master node skip connection: use only $v^t$ (Eq. (3)) without concatenating master node

<table>
<thead>
<tr>
<th>MPAD variant</th>
<th>Reut.</th>
<th>Pol.</th>
<th>IMDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPAD 1MP</td>
<td>96.57</td>
<td>79.91</td>
<td>90.57</td>
</tr>
<tr>
<td>MPAD 2MP*</td>
<td>97.07</td>
<td>80.24</td>
<td><strong>91.30</strong></td>
</tr>
<tr>
<td>MPAD 3MP</td>
<td>97.07</td>
<td>80.20</td>
<td>91.24</td>
</tr>
<tr>
<td>MPAD 4MP</td>
<td><strong>97.48</strong></td>
<td>80.52</td>
<td><strong>91.30</strong></td>
</tr>
<tr>
<td>MPAD 2MP undirected</td>
<td>97.35</td>
<td>80.05</td>
<td>90.97</td>
</tr>
<tr>
<td>MPAD 2MP no master node</td>
<td>96.66</td>
<td>79.15</td>
<td>91.09</td>
</tr>
<tr>
<td>MPAD 2MP no renormalization</td>
<td>96.02</td>
<td>79.84</td>
<td>91.16</td>
</tr>
<tr>
<td>MPAD 2MP neighbors-only</td>
<td>97.12</td>
<td>79.22</td>
<td>89.50</td>
</tr>
<tr>
<td>MPAD 2MP no master node skip connection</td>
<td>96.93</td>
<td><strong>80.62</strong></td>
<td>91.12</td>
</tr>
</tbody>
</table>

Table: Ablation results. The $n$ in $n$MP refers to the number of message passing iterations. *vanilla model.
Graph Neural Networks promising for complex Tasks
Documents represented as Graphs of Words
Message Passing GNNs for document classification tasks (MPAD)
  - Weighted, directed word co-occurrence networks,
  - MPAD is sensitive to word order and word-word relationship strength.
  - proposed hierarchical variants of MPAD, that bring improvements
THANK YOU!

ACKNOWLEDGEMENTS!
Dr. Giannis Nikolentzos

DaScIM@Ecole Polytechnique:
http://www.lix.polytechnique.fr/dascim/

Software and data sets:
http://www.lix.polytechnique.fr/dascim/software_data_sets/

We hire Ph.D.s and post-docs - contact us...