Message Passing Attention Networks for Document Understanding

Michalis Vazirgiannis

DaSciM Data Science and Mining Team (DaSciM), LIX
École Polytechnique, France and AUEB
http://www.lix.polytechnique.fr/dascim
Google Scholar: https://bit.ly/2rwmvQU
Twitter: @mvazirg

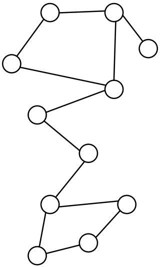
June, 2020

Talk Outline

- Introduction to GNNs
- Message Passing GNNs
- Message Passing GNNs for Document Understanding

Traditional Node Representation

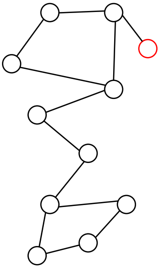
Representation: row of adjacency matrix



$$\rightarrow \begin{pmatrix} 0 & 1 & \dots & 0 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & \dots & 0 \end{pmatrix}$$

Traditional Node Representation

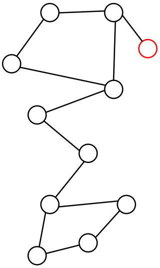
Representation: row of adjacency matrix



$$\rightarrow \begin{pmatrix} 0 & 1 & \dots & 0 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & \dots & 0 \end{pmatrix}$$

Traditional Node Representation

Representation: row of adjacency matrix



$$\rightarrow \quad \begin{pmatrix} 0 & 1 & \dots & 0 \\ 1 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & \dots & 0 \end{pmatrix}$$

However, such a representation suffers from:

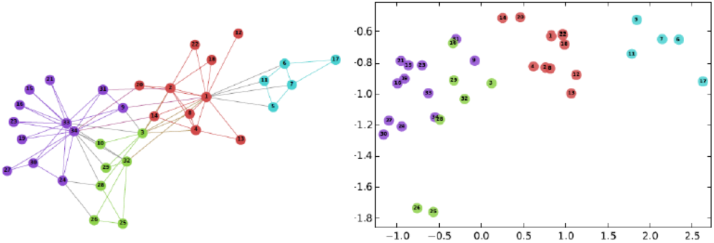
- data sparsity
- high dimensionality

$\vdots$

Node Embedding Methods

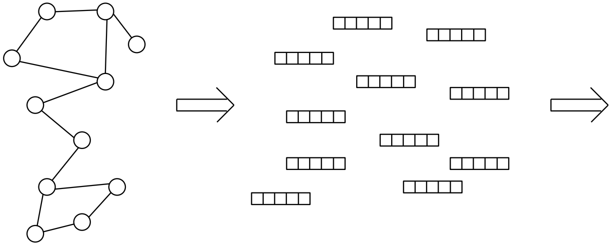
Map vertices of a graph into a low-dimensional space:

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



Input Output

Why Learning Node Representations?

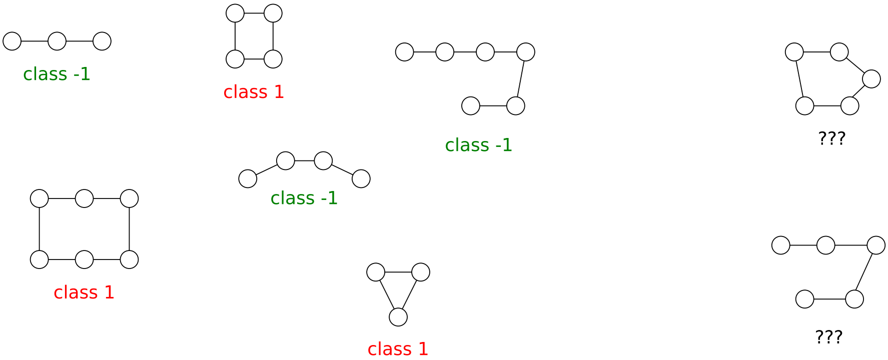


Node Classification
Anomaly Detection
Link Prediction
Clustering
Recommendation
⋮

Examples:

- Recommend friends
- Detect malicious users

Graph Classification

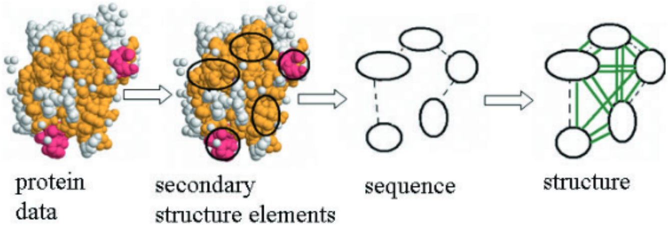


- Input data $G \in \mathcal{G}$
- Output $y \in \{-1, 1\}$
- Training set $\mathcal{S} = \{(G_1, y_1), \ldots, (G_n, y_n)\}$
- Goal: estimate a function $f : \mathcal{G} \rightarrow \in \{-1, 1\}$ to predict y from $f(G)$

Motivation - Protein Function Prediction

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

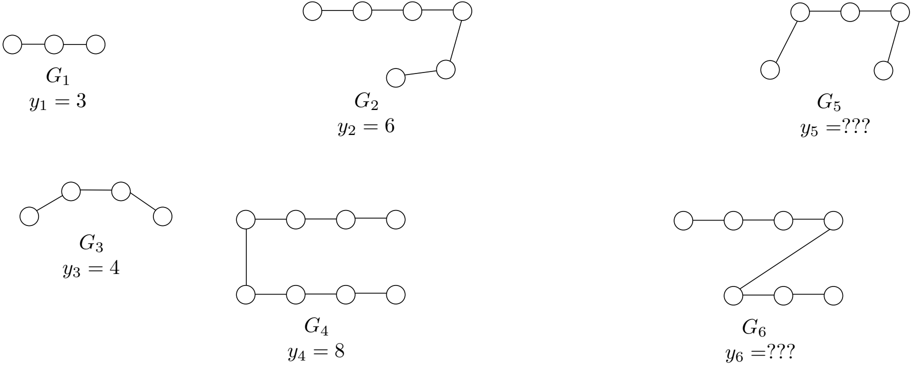
- structure
- sequence
- chemical properties



Perform **graph classification** to predict the function of proteins

[Borgwardt et al., Bioinformatics 2005]

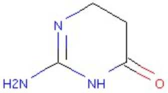
Graph Regression



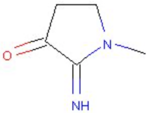
- Input data $G \in \mathcal{G}$
- Output $y \in \mathbb{R}$
- Training set $\mathcal{S} = \{(G_1, y_1), \ldots, (G_n, y_n)\}$
- Goal: estimate a function $f : \mathcal{G} \rightarrow \mathbb{R}$ to predict y from $f(G)$

Motivation - Molecular Property Prediction

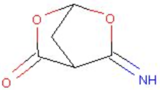
12 targets corresponding to molecular properties: ['mu', 'alpha', 'HOMO', 'LUMO', 'gap', 'R2', 'ZPVE', 'U0', 'U', 'H', 'G', 'Cv']



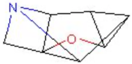
SMILES: NC1=NCCC(=O)N1
Targets: [2.54 64.1 -0.236 -2.79e-03 2.34e-01 900.7 0.12 -396.0 -396.0 -396.0 -396.0 26.9]



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]

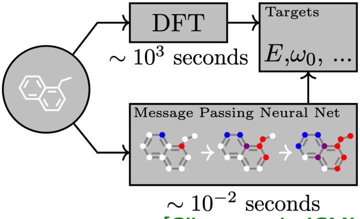


SMILES: N=C1OC2CC1C(=O)O2
Targets: [4.274 61.94 -0.282 -0.026 0.256 887.402 0.104 -473.876 -473.87 -473.869 -473.907 24.823]



SMILES: C1N2C3C4C5OC13C2C5
Targets: [? ? ? ? ? ? ? ? ? ? ? ?]

Perform **graph regression** to predict the values of the properties



[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

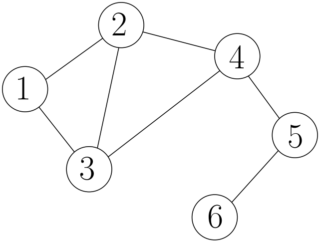
The message passing scheme runs for T time steps and updates the representation of each vertex h_v^t based on its previous representation and the representations of its neighbors:

$$m_v^{t+1} = \sum_{u \in \mathcal{N}(v)} M_t(h_v^t, h_u^t, e_{vu})$$
$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

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

$$
\begin{aligned}
\mathbf{h}_1^{t+1} &= \mathbf{W}_0^t \mathbf{h}_1^t + \mathbf{W}_1^t \mathbf{h}_2^t + \mathbf{W}_1^t \mathbf{h}_3^t \\
\mathbf{h}_2^{t+1} &= \mathbf{W}_0^t \mathbf{h}_2^t + \mathbf{W}_1^t \mathbf{h}_1^t + \mathbf{W}_1^t \mathbf{h}_3^t + \mathbf{W}_1^t \mathbf{h}_4^t \\
\mathbf{h}_3^{t+1} &= \mathbf{W}_0^t \mathbf{h}_3^t + \mathbf{W}_1^t \mathbf{h}_1^t + \mathbf{W}_1^t \mathbf{h}_2^t + \mathbf{W}_1^t \mathbf{h}_4^t \\
\mathbf{h}_4^{t+1} &= \mathbf{W}_0^t \mathbf{h}_4^t + \mathbf{W}_1^t \mathbf{h}_2^t + \mathbf{W}_1^t \mathbf{h}_3^t + \mathbf{W}_1^t \mathbf{h}_5^t \\
\mathbf{h}_5^{t+1} &= \mathbf{W}_0^t \mathbf{h}_5^t + \mathbf{W}_1^t \mathbf{h}_4^t + \mathbf{W}_1^t \mathbf{h}_6^t \\
\mathbf{h}_6^{t+1} &= \mathbf{W}_0^t \mathbf{h}_6^t + \mathbf{W}_1^t \mathbf{h}_5^t
\end{aligned}
$$

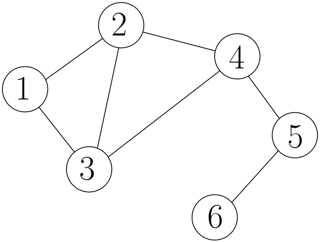


Remark: Biases are omitted for clarity

Readout Step Example

Output of message passing phase:

$$\{\mathbf{h}_1^{T_{max}}, \mathbf{h}_2^{T_{max}}, \mathbf{h}_3^{T_{max}}, \mathbf{h}_4^{T_{max}}, \mathbf{h}_5^{T_{max}}, \mathbf{h}_6^{T_{max}}\}$$



Graph representation:

$$\mathbf{z}_G = \frac{1}{6}\big(\mathbf{h}_1^{T_{max}} + \mathbf{h}_2^{T_{max}} + \mathbf{h}_3^{T_{max}} + \mathbf{h}_4^{T_{max}} + \mathbf{h}_5^{T_{max}} + \mathbf{h}_6^{T_{max}}\big)$$

Message Passing using Matrix Multiplication

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

$$\mathbf{h}_1^{t+1} = \mathbf{W}^t \mathbf{h}_1^t + \mathbf{W}^t \mathbf{h}_2^t + \mathbf{W}^t \mathbf{h}_3^t$$

- The above update scheme can be rewritten as:

$$\mathbf{h}_1^{t+1} = \sum_{i \in \mathcal{N}(v_1) \cup \{v_1\}} \mathbf{W}^t \mathbf{h}_i^t$$

- In matrix form (for all the nodes), this is equivalent to:

$$\mathbf{H}^{t+1} = (\mathbf{A} + \mathbf{I})\, \mathbf{H}^t\, \mathbf{W}^t$$

where $\mathbf{A}$ is the adjacency matrix of the graph, $\mathbf{I}$ the identity matrix, and $\mathbf{H}^t$ 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 $\mathbf{A}$ of a graph, GCN first computes the following normalized matrix:

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

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

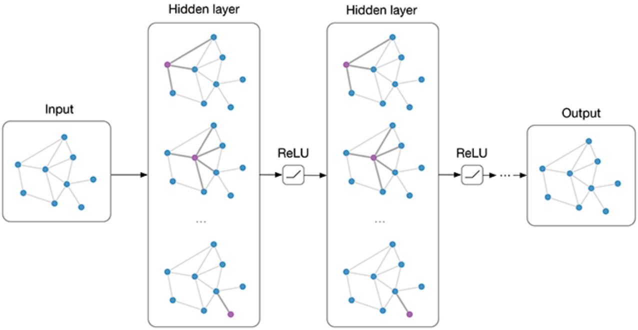
- Normalization helps to avoid numerical instabilities and exploding/vanishing gradients
- Then, the output of the model is:

$$\mathbf{Z} = \textsc{softmax}(\hat{\mathbf{A}} \; \mathrm{ReLU}(\hat{\mathbf{A}} \, \mathbf{X} \, \mathbf{W}^0) \, \mathbf{W}^1)$$

  where
  $\mathbf{X}$: contains the attributes of the nodes, i.e., $\mathbf{H}^0$
  $\mathbf{W}^0, \mathbf{W}^1$: trainable weight matrices for $t = 0$ and $t = 1$

[Kipf and Welling, ICLR'17]

GCN



To learn node embeddings, GCN minimizes the following loss function:

$$\mathcal{L} = -\sum_{i \in I} \sum_{j=1}^{|\mathcal{C}|} \mathbf{Y}_{ij} \log \hat{\mathbf{Y}}_{ij}$$

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

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

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

Results

Classification accuracies of the 4 methods

Method	Citeseer	Cora	Pubmed	NELL
DeepWalk	43.2	67.2	65.3	58.1
ICA	69.1	75.1	73.9	23.1
Planetoid	64.7 (26s)	75.7 (13s)	77.2 (25s)	61.9 (185s)
GCN	**70.3** (7s)	**81.5** (4s)	**79.0** (38s)	**66.0** (48s)

Observation: DeepWalk $\rightarrow$ unsupervised learning of embeddings

$\hookrightarrow$ fails to compete against the supervised approaches

Message Passing for document understanding

Goal: Apply the Message Passing (MP) framework to representation learning on text
$\hookrightarrow$ documents/sentences represented as word co-occurence networks

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

- documents $\rightarrow$ edge weights equal to distance between BoW representations of documents **[Henaff et al., arXiv'15]**
- documents and terms $\rightarrow$ document-term edges are weighted by TF-IDF and term-term edges by pointwise mutual information **[Yao et al., AAAI'19]**
- terms $\rightarrow$ 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-occurence 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 $\rightarrow$ unique terms
- edges $\rightarrow$ co-occurrences within a fixed-size sliding window
- vertex attributes $\rightarrow$ embeddings of terms

Graph representation more flexible than *n*-grams

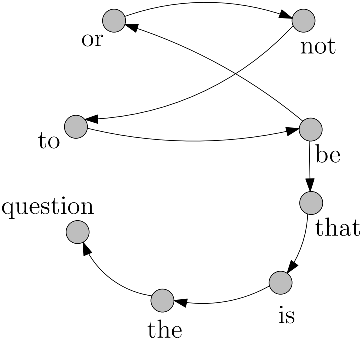


Figure: Graph representation of doc: "to be or not to be: that is the question".[Rousseau and Vazirgiannis, CIKM'13]

Message Passing Neural Networks

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

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

$$\mathbf{m}_v^{t+1} = \mathrm{AGGREGATE}^{t+1}\big(\{\mathbf{h}_w^t \mid w \in \mathcal{N}(v)\}\big)$$

The new representation $\mathbf{h}_v^{t+1}$ of v is then computed by combining its current feature vector $\mathbf{h}_v^t$ with the message vector $\mathbf{m}_v^{t+1}$:

$$\mathbf{h}_v^{t+1} = \mathrm{COMBINE}^{t+1}\big(\mathbf{h}_v^t, \mathbf{m}_v^{t+1}\big)$$

Messages are passed for T time steps

Step 2: To produce a graph-level feature vector, a $\mathrm{READOUT}$ pooling function, that must be invariant to permutations, is applied:

$$\mathbf{h}_G = \mathrm{READOUT}\big(\{\mathbf{h}_v^T \mid v \in V\}\big)$$

Message Passing Attention Networks for Document Understanding (MPAD)

- Represent textual documents as word co-occurence networks
  $\hookrightarrow$ 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

Master node: Generated networks also contain a special document node, linked to all other nodes

- can encode a summary of the document

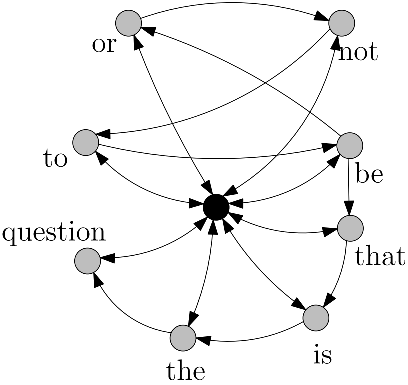


Figure: Graph representation of the document: "to be or not to be: that is the question". The black node corresponds to the **master node**

Step 1: Message Passing (1/2)

- MPAD utilizes the following AGGREGATE function:

$$\begin{aligned}\mathbf{X}^{t+1} &= \text{MLP}^{t+1}(\mathbf{H}^{t}) \\ \mathbf{M}^{t+1} &= \mathbf{D}^{-1}\,\mathbf{A}\,\mathbf{X}^{t+1}\end{aligned} \tag{1}$$

  - $\mathbf{A}$ $\Rightarrow$ adjacency matrix of word co-occurence network
  - $\mathbf{D}$ $\Rightarrow$ a diagonal matrix such that $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$
  - $\mathbf{H}^t \in \mathbb{R}^{n \times d}$ $\Rightarrow$ contains node features ($\mathbf{H}^0$ contains word (node) embeddings)
  - Renormalization $\Rightarrow$ matrix product $\mathbf{D}^{-1}\,\mathbf{A}\,\mathbf{X}^{t+1}$ computes average of neighbors' features
    $\hookrightarrow$ avoids numerical instabilities

Step 1: Message Passing (2/2)

- The COMBINE function corresponds to a GRU:

$$
\begin{aligned}
\mathbf{H}^{t+1} &= GRU(H^t, M^{t+1}) \\
\mathbf{R}^{t+1} &= \sigma(\mathbf{W}_R^{t+1}\mathbf{M}^{t+1} + \mathbf{U}_R^{t+1}\mathbf{X}^{t+1}) \\
\mathbf{Z}^{t+1} &= \sigma(\mathbf{W}_Z^{t+1}\mathbf{M}^{t+1} + \mathbf{U}_Z^{t+1}\mathbf{X}^{t+1}) \\
\tilde{\mathbf{H}}^{t+1} &= \tanh(\mathbf{W}^{t+1}\mathbf{M}^{t+1} + \mathbf{U}^{t+1}(\mathbf{R}^{t+1} \odot \mathbf{X}^{t+1})) \\
\mathbf{H}^{t+1} &= (1 - \mathbf{Z}^{t+1}) \odot \mathbf{X}^{t+1} + \mathbf{Z}^{t+1} \odot \tilde{\mathbf{H}}^{t+1}
\end{aligned}
\tag{2}
$$

  where $\mathbf{W}, \mathbf{U}$ are trainable weight matrices

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

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

Step 2: Readout

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

$$
\begin{aligned}
\mathbf{Y}^T &= \tanh(\hat{\mathbf{H}}^T \mathbf{W}_A^T) \\
\boldsymbol{\alpha}_i^T &= \frac{\exp(\mathbf{Y_i}^T \cdot \mathbf{v}^T)}{\sum_{j=1}^{n-1} \exp(\mathbf{Y_j}^T \cdot \mathbf{v}^T)} \\
\mathbf{u}^T &= \sum_{i=1}^{n-1} \boldsymbol{\alpha}_i^T \hat{\mathbf{H}}_i^T
\end{aligned}
\tag{3}
$$

Then, $\mathbf{u}^T$ is concatenated with the **master node** representation

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

$$\mathbf{h}_G = \mathrm{CONCAT}\big(\mathrm{READOUT}\big(\mathbf{H}^t\big) \mid t = 1 \dots T\big)$$

Hierarchical Variants

MPAD applied to *sentences* instead of documents → sentence representations
↪ 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

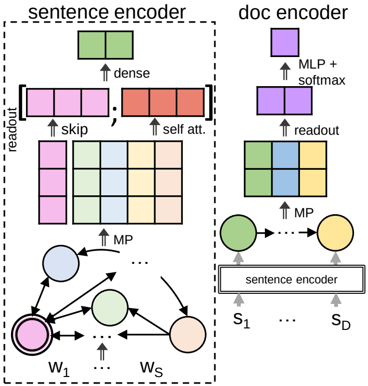


Figure: Illustration of **MPAD-path** (◎: master node).

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

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

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

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

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

Ablation Study

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

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

Table: Ablation results. The *n* in *n*MP refers to the number of message passing iterations. *vanilla model.

Conclusions

- 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

http://www.lix.polytechnique.fr/Labo/Ioannis.Nikolentzos/

DaSciM@Ecole Polytechnique:

http://www.lix.polytechnique.fr/dascim/

Software and data sets:

http://www.lix.polytechnique.fr/dascim/software_datasets/

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