GNNer: Reducing Overlapping in Span-based NER Using Graph Neural Networks

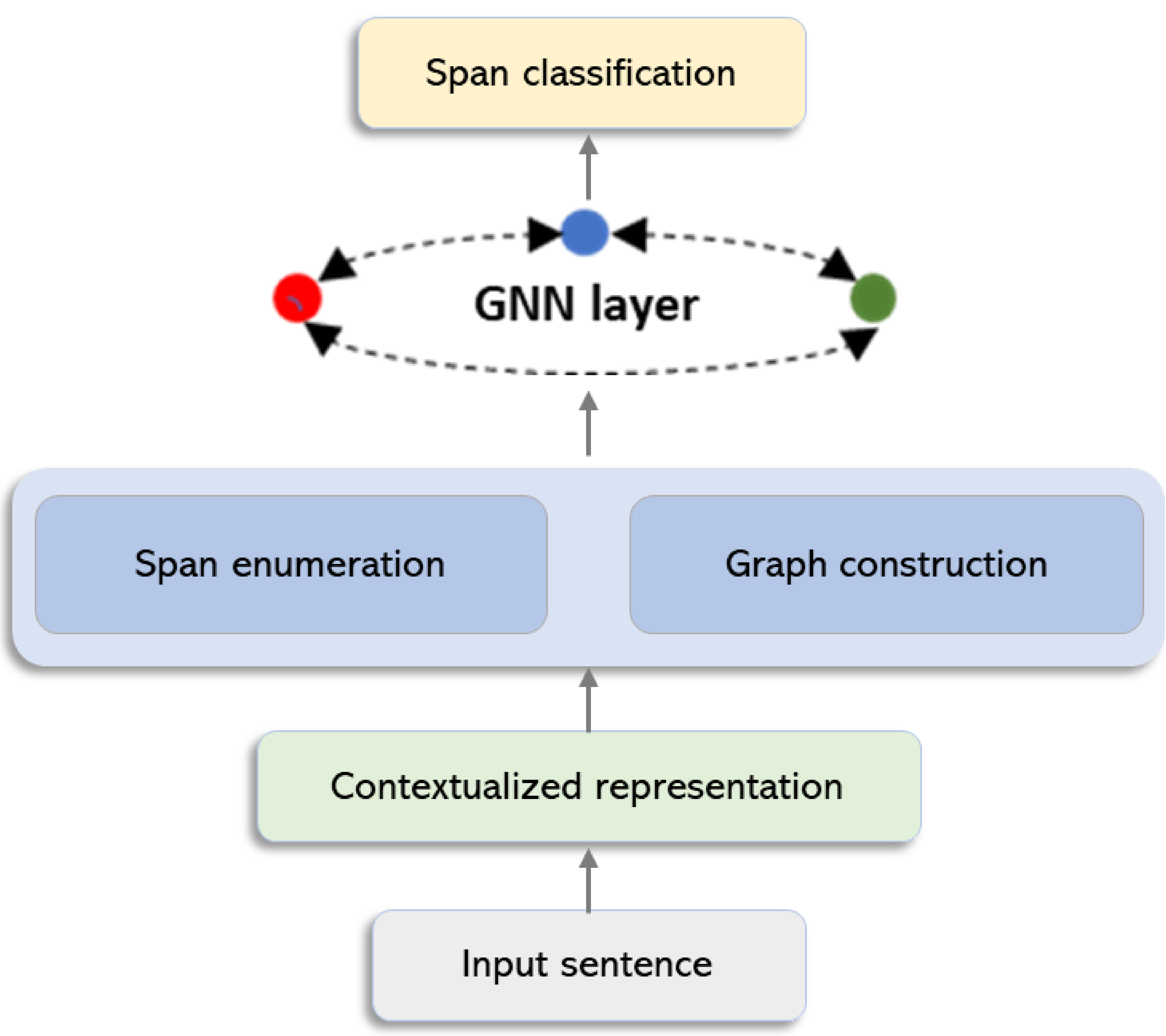
Urchade Zaratiana *‡, Nadi Tomeh ‡, Pierre Holat *‡, Thierry Charnois ‡

*FI Group, Puteaux, France

‡ LIPN, Université Sorbonne Paris Nord - CNRS UMR 7030, Villetaneuse, France

1 - Introduction

- Two main paradigms for named entity recognition: **sequence labelling** and **span classification**
  - Given a sequence of tokens
    - **X = ['Michael', 'Jordan', 'is', 'happy']**
  - *sequence labelling* performs NER by token-level prediction using scheme such as BIO (Begin Inside Outside)
    - **Y_bio = ['B-PERS', 'I-PERS', 'O', 'O']**
  - *span-based approach* enumerates all spans then classifies them
    - **Y_span = [(0, 0, 'O'), (0, 1, 'PERS'), (0, 2, 'O'), ...]**
    - Use of the span as a basic unit instead of token/word
    - Allow richer span representation compared to sequence labelling (Integrate span-level features)
- Problem: In contrast to sequence labelling, span-based approaches produce entity overlaps
  - Not suitable for flat NER tasks, i.e NER without nested entities
- Previous works employ greedy decoding to filter span overlap
  - The greedy decoding retains the span with the highest prediction probability while dropping the others
- We propose to inject span overlap information into the model architecture using Graph Neural Networks (GNN):
  - To learn span representation that is aware of overlapping neighbours
  - To bias the model towards predictions that implicitly respect the non-overlap constraints



2 - Model

- **Span graph construction**, defined by adj. matrix A

$$A[s_i, s_j] = \begin{cases} 1 \text{ if } s_i = s_j \\ -1 \text{ if } \texttt{overlap}(s_i, s_j) \\ 0 \text{ otherwise} \end{cases}$$

Negative edge for span overlap, positive egde at diagonal and no edge for no overlap

- **GNNer-CONV**

$$\mathbf{S}^+ = \texttt{GCN}_+(\mathbf{S}, E^+)$$
$$\mathbf{S}^- = \texttt{GCN}_-(\mathbf{S}, E^-)$$
$$\mathbf{S}^{final} = [\mathbf{S}^+; \mathbf{S}^-]$$

Run two independent GCNs on span representation for negative and positive edges and concatenate the results.

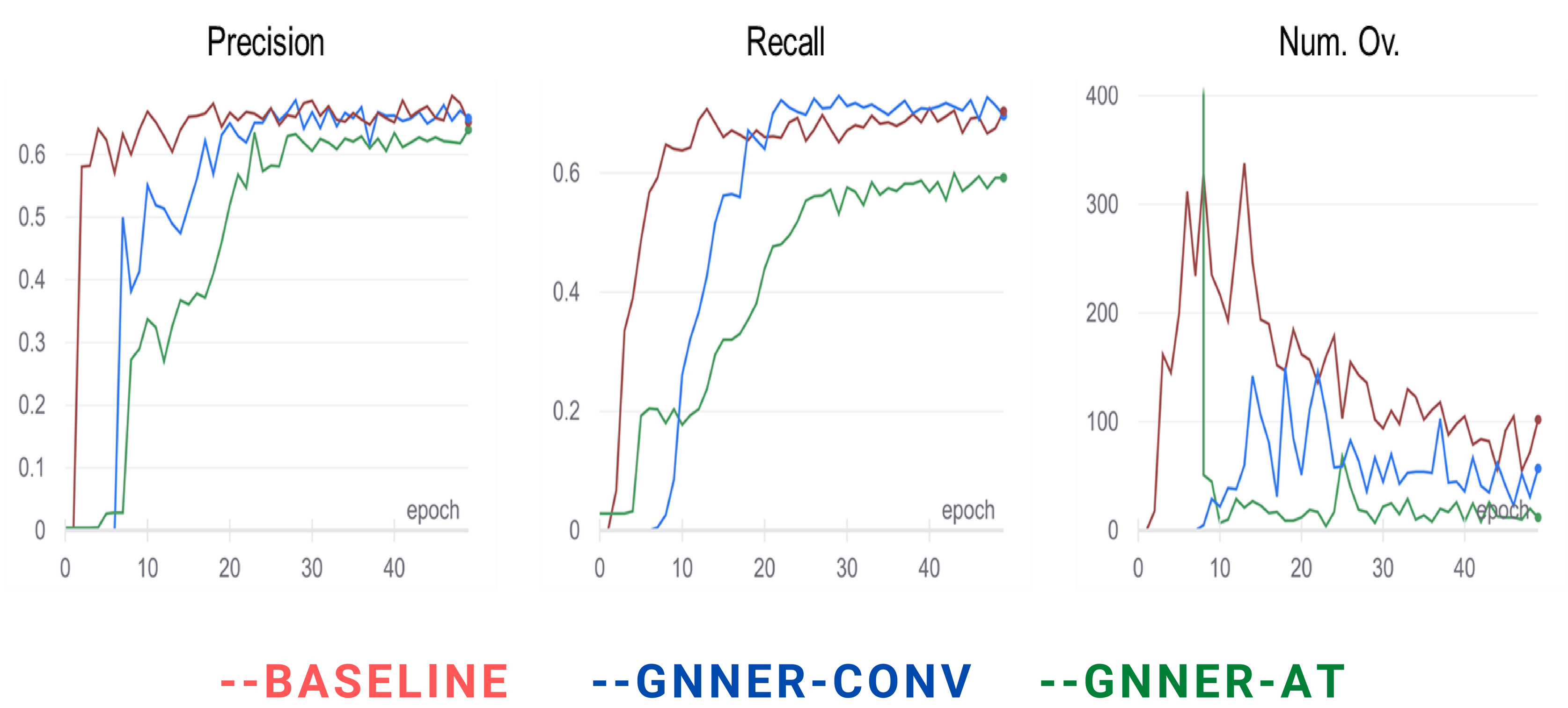
- **GNNer-AT**

$$\mathbf{S}^{final} = \left(\frac{\mathbf{QK}^T}{\sqrt{d_{model}}} \odot A \right) \mathbf{V}$$

Project span representation into **Q**, **K**, **V** and perform attention (using the adjacency matrix as mask)

3 - Experiment results

	Architecture	Precision	Recall	F1	Num. Ov.
Conll 2003	Baseline	89.83±0.48	90.31±0.26	90.06±0.15	83±27
	GNNer-CONV	90.12±0.32	89.88±0.36	**90.16±0.52**	52±1
	GNNer-AT	89.54±0.84	79.32±0.04	84.12±0.37	24±11
SciERC	Baseline	66.69±0.49	69.89±0.45	68.25±0.33	87±4
	GNNer-CONV	66.89±1.59	70.34±0.50	**68.57±0.96**	35±3
	GNNer-AT	63.21±0.51	58.06±0.86	60.53±0.69	13±2
NCBI	Baseline	85.30±0.45	89.59±0.74	87.39±0.13	43±12
	GNNer-CONV	85.98±0.45	88.93±0.45	**87.43±0.45**	16±5
	GNNer-AT	84.78±0.18	79.41±0.61	81.98±0.38	10±4



4 - Limitations and future works

- GNNs are powerful tools to integrate knowledge into deep learning models
- The model does not fully solve the overlapping span problem in contrast to heuristic approaches (the two approaches are orthogonal/can be combined together)
- Idea for future work:
  - Use soft value for edges, especially for negative edges
  - Treat positive/negative edge as edge type and use more complex GNN layer such as Heterogeneous Graph Transformer