

LLM-TO-LM INTERPRETER FOR ENHANCED TEXT-ATTRIBUTED GRAPH REPRESENTATION LEARNING



Representation Learning on TAGs

- Many real-world graphs contain textual information. These graphs are called text-attributed graphs, or TAGs.
- Representation learning on TAGs majorly focused on GNN pipelines, using shallow or hand-crafted features such as skip-grams or bag of words
- More recent works use LM (language model) based techniques, where pre-trained LMs are fine-tuned and used to generate node embeddings

Large Language Models

- For scalability reasons, LMs (like BERT and DeBERTa) must be relatively small. They thus lack the complex reasoning abilities associated with large language models
- A preliminary study found that LLMs alone can beat the accuracy of GNN-based approaches, but fall short of the state-of-the-art LM techniques.
- TAPE is an architecture that combines LLM with an LM to achieve higher accuracy

Step 1: Node Feature Extraction

Step 2: Downstream Tasks on TAGs

Prediction: cs.CV, cs.IR, cs.CL, cs.LG, cs.AI.

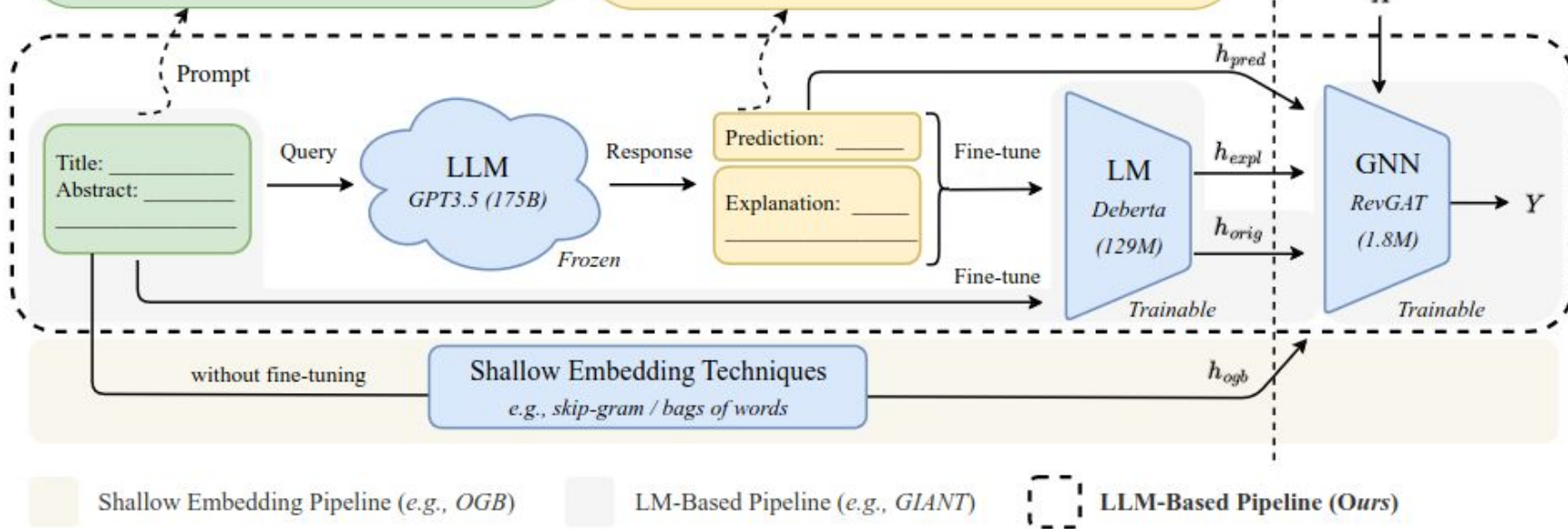
Abstract: Text in curve orientation, despite being one of the common text orientations in real world environment...

Title: Total Text A Comprehensive Dataset For Scene Text Detection And Recognition.

Question: Which arXiv CS sub-category does this paper belong to? Give 5 likely arXiv CS sub-categories as a comma-separated list ordered from most to least likely, in the form "cs.XX", and provide your reasoning.

Answer:

Explanation: The paper is about a new dataset for scene text detection and recognition, which is a topic related to computer vision (cs.CV). The paper also mentions the use of deep learning techniques such as DeconvNet, which falls under the sub-category of artificial intelligence (cs.AI). The dataset is annotated and involves text recognition, which could also fall under the sub-categories of information retrieval (cs.IR) and natural language processing (cs.CL). Finally, the paper discusses the effectiveness of different solutions, which could be evaluated using machine learning techniques, falling under the sub-category of machine learning (cs.LG).



Abstract: [paper abstract]

Title: [paper title]

Question: [ask the model to predict one or more class labels of the paper, ordered from most to least likely, and provide explanations for its predictions]

Answer:

Querying the LLM results in a ranked prediction list and a textual explanation for each paper:

(Ranked Predictions) [a ranked prediction list]

(Explanations) [model-generated explanation for the predictions]

Concretely, we first fine-tune pre-trained LMs as follows: let LM_{orig} and LM_{expl} be pre-trained LMs that take as input the original s^{orig} and the explanation s^{expl} text sequences, respectively. We obtain text embeddings for each source as follows:

$$h_{\text{orig}} = \text{LM}_{\text{orig}}(s^{\text{orig}}) \in \mathbb{R}^{N \times d}, \quad h_{\text{expl}} = \text{LM}_{\text{expl}}(s^{\text{expl}}) \in \mathbb{R}^{N \times d}. \quad (4)$$

We further apply a Multi-Layer Perceptron (MLP) to the output of the LMs to obtain a $N \times C$ -dimensional prediction matrix representing the LM's predictions for each node (in logits):

$$y_{\text{orig}} = \text{MLP}_{\text{orig}}(h_{\text{orig}}) \in \mathbb{R}^{N \times C}, \quad y_{\text{expl}} = \text{MLP}_{\text{expl}}(h_{\text{expl}}) \in \mathbb{R}^{N \times C}. \quad (5)$$

We fine-tune these LMs and MLPs using cross-entropy loss. Finally, the text embeddings from both sources, h_{orig} and h_{expl} , are used as enriched features for training downstream GNNs.

Ranked prediction features. In addition to the explanations, the LLM also provides a top- k ranked prediction list for each node, which adds valuable information. To incorporate this knowledge, the top- k predictions for node i are first one-hot encoded as vectors $p_{i,1}, \dots, p_{i,k} \in \mathbb{R}^C$. These vectors are subsequently concatenated into a kC -dimensional vector, followed by a linear transformation to produce a fixed-sized vector of length d_P . This process produces a prediction feature matrix as $h_{\text{pred}} \in \mathbb{R}^{N \times d_P}$ across all nodes.

Advantages

- State-of-the-art performance, with lesser computation time than the previous SOTA
- LMaaS-compatible: This approach is compatible with language-model-as-a-service offerings such as GPT-3.5

Comparison with previous SoTA: GLEM

- GLEM uses a multi-stage iterative approach where LM and GNN are alternatively trained. TAPE trains both separately

Table 2: Experiments on `ogbn-arxiv` dataset with DeBERTa-base (He et al., 2021) as LM backbone and RevGAT (Li et al., 2021) as GNN backbone for comparison of different training paradigms of fusing LMs and GNNs, including our proposed method and the state-of-the-art GLEM method (Zhao et al., 2022). The validation and test accuracy, number of parameters, maximum batch size (Max bsz.), and total training time on 4 NVIDIA RTX A5000 24GB GPUs are reported.

Method	Val acc.	Test acc.	Params.	Max bsz.	Total time
LM _{orig}	0.7503 \pm 0.0008	0.7361 \pm 0.0004	139,223,080	36	1.73h
GNN- h_{shallow}	0.7144 \pm 0.0021	0.7083 \pm 0.0017	427,728	all nodes	1.80min
GLEM-G-Step	0.7761 \pm 0.0005	0.7657 \pm 0.0029	1,837,136	all nodes	9.18h
GLEM-L-Step	0.7548 \pm 0.0039	0.7495 \pm 0.0037	138,632,488	36	
TAPE-LM _{orig} -Step	0.7503 \pm 0.0008	0.7361 \pm 0.0004	139,223,080	36	1.73h
TAPE-LM _{expl} -Step	0.7506 \pm 0.0008	0.7432 \pm 0.0012	139,223,080	36	1.40h
TAPE-GNN- h_{TAPE} -Step	0.7785 \pm 0.0016	0.7750 \pm 0.0012	1,837,136	all nodes	3.76min

Results obtained using an MLP classifier over BERT, Deberta and Sentencebert: From the ENGINE paper, TAPE paper, and on local machine

	BERT	DeBERTa	SentenceBERT
ENGINE (table 2)	80.15 ± 1.67	77.79 ± 2.26	78.82 ± 1.39
TAPE (table 1)	-	76.06 ± 3.78	-
experimental	79.41 ± 1.54	78.38 ± 0.96	79.56 ± 1.54

All numbers within margin of error, variance explained by random sampling

Code, Environment, Hyperparameters

https://github.com/ode/finetune_lm

- Results obtained on NVIDIA RTX A5000, 24GiB vRAM
- Same hyperparameters as used in ENGINE for BERT and SentenceBert
- Learning rate of $2e-5$ and weight decay 0 used for Deberta (taken from TAPE)