ConGraT: Self-Supervised Contrastive Pretraining for Joint Graph and Text Embeddings

William Brannon* **Suyash Fulay** **Hang Jiang** **Wonjune Kang**

Brandon Roy **Jad Kabbara** **Deb Roy**

MIT Center for Constructive Communication
{wbrannon, sfulay, hjian42, wjkang,
bcroy, jkabbara, dkroy}@mit.edu

Abstract

We propose ConGraT[1] (**Con**trastive **Gra**ph-**T**ext pretraining), a general, self-supervised method for jointly learning separate representations of texts and nodes in a parent (or "supervening") graph, where each text is associated with one of the nodes. Datasets fitting this paradigm are common, from social media (users and posts), to citation networks over articles, to link graphs over web pages. We expand on prior work by providing a general, self-supervised, joint pretraining method, one which does not depend on particular dataset structure or a specific task. Our method uses two separate encoders for graph nodes and texts, which are trained to align their representations within a common latent space. Training uses a batch-wise contrastive learning objective inspired by prior work on joint text and image encoding. As graphs are more structured objects than images, we also extend the training objective to incorporate information about node similarity and plausible next guesses in matching nodes and texts. Experiments on various datasets reveal that ConGraT outperforms strong baselines on various downstream tasks, including node and text category classification and link prediction.

1 Introduction

Recent advances in multimodal representation learning have demonstrated the benefits of simultaneously modeling language and another modality, which allows for more efficient training and improved performance of both sets of learned representations on downstream tasks. These benefits have been particularly clear in text/vision or text/audio applications, which often realize large improvements in predictive performance or generative modeling ability (Elizalde et al., 2019; Radford et al., 2021; Li et al., 2021; Mu et al., 2022). In this work, however, we consider another modality which frequently co-occurs with text: network- or graph-structured data.

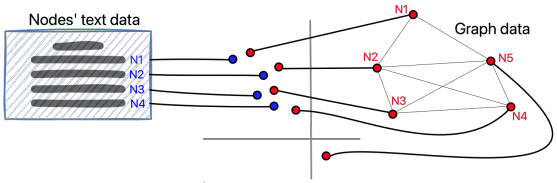

Figure 1: Embeddings of graph nodes (red) and their associated texts (blue). They are placed into a common embedding space, in which nodes are embedded near their associated texts. Node-text pairs are labeled N1 to N5. Note that not every node must have an associated text (here, N5 does not).

We consider in particular the scenario in which there is a graph over entities which generate or contain texts, and in which each text is associated with a node in the graph. Such graphs co-occur frequently with real text data, and we refer to them as "supervening graphs" over their text corpora. Examples include tweets and the follow graph over the posting Twitter users, web page content and the link graph over the pages, the text of academic articles and the citation network over articles or authors, and others. In this setting, graph information can be used to improve performance on language tasks, and text features may be leveraged for graph tasks, such as link prediction or node classification.

Prior work has approached the problem of taking joint advantage of these two modalities in several ways. Some work has used textual data to inform or supervise training of graph neural networks (Zhang and Zhang, 2020; Zhang et al., 2017), including seq2seq approaches for node representations (Liu et al., 2018), the use of text features in DeepWalk-style algorithms (Yang et al., 2015), and kernel methods to incorporate text into node representations (Zhang et al., 2017). None of these approaches, however, also produce graph-informed text representations. This is more

*Corresponding author.

[1]Code and certain datasets are available at https://github.com/wwbrannon/congrat.

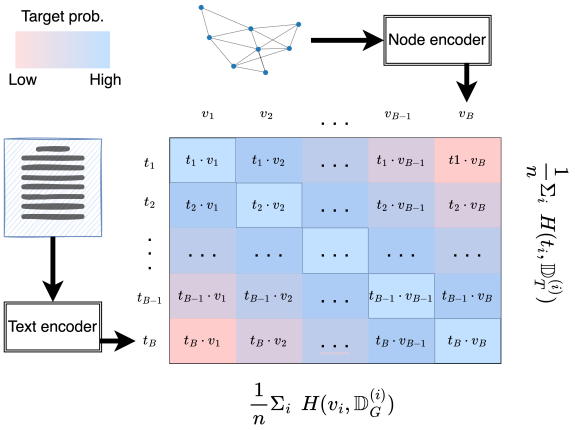

Figure 2: The overall architecture of our model. Given a minibatch of (text, origin node) pairs, node and text embeddings are generated by their respective encoders, then used to compute pairwise cosine similarities. The final loss is the average of cross entropies along each row and column of the similarity matrix, with each row i's target probabilities (labeled $\mathbb{D}_T^{(i)}$ and $\mathbb{D}_G^{(i)}$) a mixture of the true targets (on the diagonal) and a (row- or column-specific) distribution proportional to a graph-based similarity measure.

parameter-efficient for graph-only tasks, but the common practical case of also needing to solve text-based tasks then requires an additional modeling approach be used to leverage graph data.

Other work has considered the converse case of employing a supervening graph structure to fine-tune a language model. LinkBERT (Yasunaga et al., 2022) constructs fine-tuning examples based on inter-document links, while SPECTER (Cohan et al., 2020) and SciNCL (Ostendorff et al., 2022) use contrastive objectives to update a pretrained language model. Although these approaches allow for the extraction of individual document representations in the form of sentence embeddings, they have a parallel limitation to graph-focused models in that they do not learn node representations.

While there have been some attempts to jointly represent nodes and texts, they all have certain limitations. For example, Chandra et al. (2020) and Li and Goldwasser (2019) jointly train text and graph encoders, but require a supervised objective and labeled data. Karpov and Kartashev (2022) allow a language model to attend over graph embeddings, but the training process does not further train or update the graph embeddings. Gourru et al. (2020), similarly, learn to embed documents into a "pretrained semantic space" but the semantic embedding model is not updated in light of graph information. Li et al. (2017), while conceptually similar to our work, carefully curate their input datasets and rely on particular structure and node attributes found in social network discourse. None of the prior joint representation approaches we are aware of are suitable for general, self-supervised, joint pretraining without dependence on labeled data or task structure.

In this work, we propose ConGraT (**Con**trastive **Gra**ph-**T**ext pretraining), a general approach to self-supervised joint graph-text learning, based on a batch-wise contrastive learning objective inspired by InfoNCE (Oord et al., 2019) and CLIP (Radford et al., 2021). The idea is to have separate encoders for the language and graph modalities that are trained to align their representations within a common latent space, as shown in Figure 1. Because graphs are more structured objects than images, we are able to modify the InfoNCE objective to incorporate information about plausible "next guesses" based on graph similarity, and observe improvements in performance as a result. This approach provides flexibility in the choice of text and graph encoders (parameter count, architecture, pretraining, etc.), and is inductive (Hamilton et al., 2017), with the encoders able to generalize to previously unseen graphs as well as previously unseen texts. The overall architecture is shown in Figure 2.

Experiments on various datasets show that ConGraT models consistently outperform strong baselines on a number of tasks. We see statistically significant improvements on node category classification in 25 out of 36 experiments, with neither our model nor any baseline performing significantly better in a further 8. Furthermore, when applied zero-shot to link prediction, ConGraT models achieve better performance on two of three datasets compared to a graph model specifically trained to do the task. We also examine whether pretraining hurts downstream language modeling performance, finding at worst a slight reduction and often no statistically significant change.

The contributions of this work are threefold. First, we propose a general contrastive pretraining method for text corpora accompanied by a supervening graph, such as a follow, link, or citation graph. The pretraining produces separate text and node encoders targeting a common latent space, and, importantly, is both inductive and self-supervised. Second, we demonstrate that our joint pretraining method improves performance on various downstream tasks over strong unimodal and

cross-modal baselines. Finally, we release our code and datasets, including in particular a version of the Pubmed (Sen et al., 2008) graph learning benchmark fully rebuilt from ground-truth Pubmed APIs, which includes the text of titles and abstracts as well as network data. We hope these datasets will be useful to multiple research communities working on joint text-graph problems.

2 Related Work

Text-augmented GNN training. Initial node representations in inductive graph models are often created from the text associated with each node, as in the bag-of-words vectors employed in the popular Pubmed (Sen et al., 2008), Cora (McCallum et al., 2000), and Citeseer (Giles et al., 1998) benchmarks. Text data is also incorporated into node representations in other ways: for example, Yang et al. (2015) extend the DeepWalk algorithm to incorporate text features into node representations; Liu et al. (2018) learn node embeddings in a seq2seq framework with inputs based on texts associated with the nodes; Tu et al. (2017) use a selective attention mechanism to generate text-informed node embeddings for particular social contexts. These models do not also learn graph-informed text representations. Zhang et al. (2017) leverage kernel methods to construct node representations from user profile information in a way that incorporates network structure. Other approaches include extracting graphs from entity co-occurrence in texts and modeling them (Zhang and Zhang, 2020; Waller and Anderson, 2021).

Graph-augmented LM training. Several recent works have used hyperlinks or citation graphs to inform language model (LM) training. SPECTER (Cohan et al., 2020) contrastively fine-tunes a language model to produce document embeddings: the positive and negative examples for a triplet loss are selected according to citation graph edges. LinkBERT (Yasunaga et al., 2022) uses citation or link graphs to assemble training samples for a masked language model, pairing anchor texts with texts from the same contiguous document, linked documents, or random documents. In addition to the standard masked language modeling training objective, it uses an auxiliary document relation prediction (DRP) objective, which classifies the relation of the two text segments in the input. SciNCL (Ostendorff et al., 2022) relaxes a discrete citation graph into a continuous domain with nearest-neighbor sampling. Unlike in the joint-embedding case, these models represent documents or authors not directly, but as pooled LM outputs.

Joint graph-text representations. Prefix tuning (Li and Liang, 2021) is a lightweight way of learning node-specific linguistic information and generates dense node representations in the process. However, it takes no advantage of the graph structure over the nodes. For fixed text and graph encoders, it is possible to learn mappings from their separate embedding spaces to a common one, such as by canonical correlation analysis (Gupta and Varma, 2017).

Chandra et al. (2020) jointly train both text and graph encoders, but use an externally supervised objective such as fake news detection. Li and Goldwasser (2019) consider the task of predicting news stories' political perspectives, employing both text and social graph encoders trained with a story-wise, non-contrastive embedding alignment term added to a supervised loss. Gourru et al. (2020) learn to embed documents connected by links into a pretrained semantic space, taking care to represent uncertainty in the embeddings, but the setting is less general than the one we consider here (only one text document is associated with each graph node), and the pretrained semantic space is frozen.

Other works have attempted to condition a language model or augment its input with dense representations of graph entities. SocialBERT (Karpov and Kartashev, 2022) and LMSOC (Kulkarni et al., 2021) inject social information into language models during training via fixed graph node representations that the model can attend over. Li et al. (2022) use knowledge-graph representations of Twitter user attributes to inform language modeling on tweets. Li et al. (2017) learn joint embeddings of social network posts and users for general predictive tasks, with users represented initially by attribute vectors. This work, while similar to ours in spirit, has several differences from our approach. Training involves (positive, negative) pairs of examples rather than contrastive learning at the minibatch level, as we use. More notably, training relies on structure particular to certain kinds of online interactions and on access to rich user attributes, making it inapplicable as a general baseline.

3 Problem Formulation

We consider a directed or undirected graph over a set of nodes, where each node is associated with

a set of one or more texts. The goal is to learn a common latent space that allows us to compare the embeddings of nodes and texts by placing them in meaningful locations within that space. For example, the nodes and texts may be users in a social network and their posts, websites and their pages, authors and their articles, etc.

Formally, let $G = (V, E)$ be a graph, with V the set of nodes and $E \subseteq V \times V$ the set of edges. Also, let $T^{(v)} = \{t_i^{(v)}\}_{i=1}^{N_v}$, for $v \in V$, be the set of node v's texts, with N_v the number of texts corresponding to node v. We model $t_i^{(v)}$, the i-th text of node v, as a finite sequence of tokens over a vocabulary W, where $L_i^{(v)}$ is the length of $t_i^{(v)}$: $t_i^{(v)} = (S_0, S_1, S_2, ..., S_{L_v^{(i)}})$. The first and last tokens are always special start and end tokens.

Our training framework involves a *text encoder*, a function $F_T : \cup_{i=1}^{\infty} \otimes_i W \to \mathbb{R}^d$ from the set of all token sequences to a d-dimensional Euclidean embedding space. Similarly, we have a *node encoder*, a function $F_G : V \to \mathbb{R}^d$ to an embedding space of the same dimension. We aim to train the two encoders such that they directly learn a joint latent space between the text and graph node embeddings. This will allow us to use geometric properties of a common space to relate nodes and texts to each other for predictive or inferential purposes.

4 Approach

Our approach involves two parallel blocks (separate text and node encoders) connected at the output layers by a loss function, specifically a batch-wise contrastive objective inspired by InfoNCE (Oord et al., 2019) and CLIP (Radford et al., 2021). The training process induces the encoders, which do not in general target the same embedding space, to align their representations. This approach is flexible and allows for the use of many kinds of both text and node encoders. On the text side, we illustrate this flexibility with experiments employing both causal and masked language models.

As in CLIP, each encoder is placed behind an adapter module which generates embeddings of the same dimension, since we would like to embed nodes and texts in the same space. Our adapters consist of two fully connected layers, with a GeLU (Hendrycks and Gimpel, 2020) in between. The adapter is followed by layer normalization (Ba et al., 2016) and dropout (Srivastava et al., 2014).

Training objective. We modify the standard InfoNCE loss with additional graph-specific elements. Unlike the vision-language case, in a graph setting, there are easily computable measures of how similar pairs of nodes are, such as their SimRank (Jeh and Widom, 2002) or the number of mutual in- and out-edges. We exploit these measures to incorporate information about the most likely second, third, and further choices for the nodes a text came from and the texts a node produced.

More formally, in terms of the notation in Section 3, let $X = \cup_{v \in V} \{(v, t_i^{(v)})\}_{i=1}^{|T^{(v)}|}$ be a dataset of (node, text) pairs, and let $B = \{(v_i, t_i)\}_{i=1}^{N_B} \subseteq X$ be a minibatch of size N_B sampled from X. We drop the (v) superscript for simplicity. Using the text and node encoder notation from Section 3, $F_T(t_i)$ is the text encoder output vector for text i, and $F_G(v_i)$ is node v's graph encoder output.[2] Then the matrix C given by

$$C_{ij} = \frac{F_T(t_i) \cdot F_G(v_j)}{\|F_T(t_i)\| \cdot \|F_G(v_i)\|} \, e^{\tau} \tag{1}$$

is the $N_B \times N_B$ matrix of cosine similarities between texts and nodes. (Note that, though square, this matrix is not symmetric: rows are texts and columns are nodes.) We multiply each element by e^{τ}, where τ is a log-temperature parameter that allows a degree of learnable control over the learning rate, reducing the model's sensitivity to the choice of LR. Further, let $S_T(\cdot, \cdot)$ and $S_G(\cdot, \cdot)$ be non-negative graph-based *similarity functions* for texts and nodes, respectively. Then we define the graph-based similarity distributions for texts and nodes as follows:

$$s_T^{(i)}(j) = \frac{S_T(t_i, t_j)}{\sum_{k=1}^{N_B} S_T(t_i, t_k)} \; \forall i \tag{2}$$

and analogously for $s_G^{(i)}$, replacing S_T with S_G and t_i with v_i. The target distributions are mixtures of these distributions and indicator variables for the true source node of a text and matching text of a node. For each example $X_i = (v_i, t_i)$ in the minibatch, fixing some hyperparameter $\alpha \in [0, 1]$, we define the target distributions as follows:

$$\mathbb{D}_T^{(i)}(\alpha) = (1 - \alpha)\mathbb{1}_j\{v_j = v_i\} + \alpha \vec{s_T}(t_i), \tag{3}$$

$$\mathbb{D}_G^{(i)}(\alpha) = (1 - \alpha)\mathbb{1}_j\{t_j = t_i\} + \alpha \vec{s_G}(v_i). \tag{4}$$

[2] Concretely, the inputs to these encoders are a token sequence and attention mask, on the text side, and an initial vector representing the node, on the graph side.

Then our loss function is:

$$\mathcal{L}(B;\alpha) = \frac{1}{2N_B}\sum_{i=1}^{N_B} H(C_{i,:}, \mathbb{D}_T^{(i)}(\alpha)) + H(C_{:,i}, \mathbb{D}_G^{(i)}(\alpha)) \quad (5)$$

where H indicates the cross-entropy and $C_{i,:}$, $C_{:,i}$ are the i-th row and i-th column of C.

With $\alpha = 0$, this loss is equivalent to the average of cross-entropies for predictions of which node in the minibatch goes with which text and which text goes with which node. With higher values of α, the target distributions are mixtures of: a) indicators for the true source node and text, and b) the distribution of other nodes and texts by graph similarity. If similar nodes produce similar texts, as happens in many real social networks (De Choudhury et al., 2010), positive values of α should allow the model to learn more information and do so more efficiently.

Similarity function. For undirected graphs, we base our similarity function on the number of mutual neighbors of a pair of nodes. If A is the graph adjacency matrix, we compute AA^T to find the number of in- or out-neighbors of each pair of nodes, and find the cosine similarity of each row i of AA^T with each column j to measure the similarity of nodes i and j.[3] We also evaluated SimRank (Jeh and Widom, 2002) as the similarity metric, but opted for this function instead for its lower computational requirements and faster runtime for large graphs. On the text side, because we're interested in leveraging graph information, we proxy for the similarity of a pair of texts with the similarity of the nodes which produced them.

The digraph case is more complicated, as we need a directed notion of similarity that can differ between edges (i, j) and (j, i). We elect not to choose and validate such a similarity function here, deferring the question to future work. Accordingly, experiments with $\alpha > 0$ in Section 5 and Section 6 all discard edge directions.

5 Experiments

5.1 Datasets

We employ three datasets, one each from social-, citation-, and link-graph domains. We divide each dataset into a 70% train set, 10% validation set, and 20% test set, splitting at the node level so that every text associated with a given node is in the same data split. On the graph side, any edges which cross split boundaries are dropped. Notably, because how to scale graph neural networks to very large graphs is still an active research area (Serafini and Guan, 2021; Ma et al., 2022), our datasets feature at most tens of thousands of nodes and a few million edges. Details of the datasets can be found in Appendix A, including Table 4 and Table 5.

Pubmed. We built from scratch a version of the popular Pubmed graph learning benchmark (Sen et al., 2008) which includes the titles and abstracts of each article (widely available versions of the dataset do not include any text). We began with the standard list of PMIDs for the articles in the dataset, and fetched from the Pubmed API the title, abstract, and list of references. We kept directed citation edges only to other articles in the dataset. One PMID was not found in the Pubmed database and was thus left out. The final dataset includes 19,716 nodes, 61,110 edges, and 59,381 texts, including both titles and abstracts. The included articles are about diabetes, and the standard node categories, which we use in our experiments, are from the Pubmed database: type-1 diabetes, type-2 diabetes, or experimental evidence.

T-REx. We use a dataset of Wikipedia articles, represented by their introductory paragraphs and the hyperlinks between the articles in those paragraphs, selected from the broader T-REx corpus (Elsahar et al., 2018) of Wikipedia articles and their knowledge-base triples. Rather than use the entire corpus, we selected a sample to satisfy computational constraints, choosing a single category arbitrarily such that it resulted in a connected graph of about the same size as the other datasets. The final dataset includes 9,214 nodes (articles), 22,689 edges, and 18,422 texts, including the titles and initial paragraphs of the articles. A total of 1,433 unique subcategories (descending from the parent category) are associated with these articles. We derived five binary label sets over the documents from the subcategories, with each consisting of a group of related subcategories. Further details of label construction are given in the appendix.

Twitter. We create and use a Twitter dataset of 167,558 tweets posted by a set of 8,721 influential users in media, politics, and entertainment, and the follow graph among these users, which consists of 2,373,956 edges. We include in the dataset up to

[3] This is one of several common simple ways to measure node similarity; see for example (Salton, 1988).

20 tweets per user, sampled from each user's most recent 3,200 tweets as of May 9, 2021. We also collected certain demographic data about these users (region of residence, age, gender, and politician vs. entertainer occupation status) by matching them with information on Wikipedia and Ballotpedia.

5.2 Models

Our experiments cover and explore three choices left open by the framework described in Section 4: whether to use a causal or masked language model as a text encoder, whether to use graph-derived similarity information in the loss (specifically with $\alpha = 0.1$) or not ($\alpha = 0$), and whether to consider edge directions. On each dataset, we train ConGraT models for 6 of the 8 combinations of these factors, four discarding edge directions and two keeping them, for a total of 18 models over all datasets. (Recall from Section 4 that we defined a similarity function to use for $\alpha > 0$ only for undirected graphs, so there are neither causal nor masked LM experiments with directed edges and $\alpha > 0$.)

All experiments use transformer LMs for the text encoders, although our approach is generalizable to other architectures as well. For masked LM experiments, we initialize weights with the pretrained `all-mpnet-base-v2` model (Song et al., 2020) from the sentence-transformers toolkit (Reimers and Gurevych, 2019), while our causal LM experiments initialize weights with the pretrained `distilgpt2` model (Sanh et al., 2019). The token-level outputs of the text encoders are mean-pooled to obtain text-level representations. For the graph node encoder, we use a graph attention network (GAT) (Veličković et al., 2018) with 3 layers and 2 attention heads in each layer for all experiments, randomly initialized and trained from scratch. All text and graph embeddings are set to have dimensions of 768. Further details of these architectures and training are provided in Appendix B.

5.3 Evaluation

We evaluate our approach in three ways: link prediction, language modeling, and node category classification. Link prediction and language modeling are used as fundamental modality-specific metrics that are meant to measure how well the node and text encoders from our method maintain their abilities to model their individual modalities. We perform node category classification using each encoder's embeddings in order to measure how effective the learned representations are for downstream tasks. We describe the various baselines we used in these experiments below; their architectures and training details are described in Appendix B.

Link prediction. We evaluate how well the graph encoders trained using ConGraT perform at link prediction, the task of predicting whether a pair of nodes is connected by an edge. We perform link prediction via inner product decoding (Kipf and Welling, 2016); given node embeddings, we form the pairwise matrix of inner products, apply a sigmoid function element-wise, and take the resulting values as probabilities that edges exist. As a baseline, we use the same GAT architecture as in our jointly trained ConGraT models and train it on a graph autoencoding task (Kipf and Welling, 2016) using the inner product decoding scheme described above. Initial node representation vectors are based on the truncated SVD of the data split-specific adjacency matrix.

Language modeling. After joint training, the node and text encoders trained using ConGraT are directly usable for tasks which require node or text embeddings, but the text encoder cannot be immediately used for language modeling because it lacks an LM head. To evaluate the impact of joint pretraining on downstream language-modeling performance, we attach a randomly initialized LM head to the jointly trained text encoder and further train it to perform causal language modeling. We use perplexity as our metric; thus, we limit our evaluation here to the causal LM (`distilgpt2`) variant of our model with $\alpha = 0$. As our baseline, we fine-tune a unimodally pretrained `distilgpt2` to perform causal language modeling in the same way.

Category classification. To evaluate the usefulness of our learned representations on downstream tasks, we perform node category classification on each dataset. For Pubmed, the classes are the type-1/type-2/experimental labels provided with the dataset; for T-REx, the Wikipedia-derived article categories; and for Twitter, the Wikipedia- and Ballotpedia-derived demographic labels. Within the test split of each dataset, we create another 50/50 train/test split of the data at the node level, then train logistic regression models to predict the aforementioned categories using either node or text embeddings. Specifically, we freeze the text and node encoders in our model and use each modality's representations as input features for logistic regression. When using text embeddings, since it

	Undirected			Directed		
	Pubmed	T-REx	Twitter	Pubmed	T-REx	Twitter
Unimodal GAT	0.871	0.832	0.886	0.908	0.699	0.843
ConGraT-Causal ($\alpha = 0.0$)	**0.976**	**0.886**	0.793	**0.952**	**0.755**	0.819
ConGraT-Masked ($\alpha = 0.0$)	**0.984**	0.816	0.805	**0.955**	0.660	0.826
ConGraT-Causal ($\alpha = 0.1$)	**0.983**	**0.932**	0.785	–	–	–
ConGraT-Masked ($\alpha = 0.1$)	**0.985**	**0.940**	0.790	–	–	–

Table 1: Jointly trained graph encoders' performance at link prediction, compared against a baseline GAT trained on a graph autoencoding task. Values shown are AUCs for the binary classification problem of predicting whether each edge in the test-set graph exists. Note that these results are inductive, with the test-set graph not available at training time. Bolded values indicate improved performance over the baseline.

is possible for each node to be associated with multiple texts (e.g., a Twitter user may have multiple tweets), we use the average of the text embeddings associated with each node to predict the node's various characteristics. We also test a combined embedding using both modalities, in which the graph and text embeddings from both encoders of our model are concatenated to create a single embedding of dimension 1536 that is then used to predict the node class.

We compare against several baselines. For the node representations, we compare against embeddings from the separately trained GAT autoencoder that was also used for link prediction. For text representations, in addition to unimodal masked and causal LM baselines that were fine-tuned on each dataset, we also compare against two graph-augmented LMs: Social-LM, a modified implementation of SocialBERT (Karpov and Kartashev, 2022) and LMSOC (Kulkarni et al., 2021), and LinkBERT (Yasunaga et al., 2022). To compare against the combined graph and text embeddings, we combine each of the text baselines with the GAT graph baseline by concatenating their embeddings together in the same way as we do for our method.

6 Results

6.1 Link Prediction

As shown in Table 1, ConGraT graph node encoders deliver strong performance on link prediction. On the Pubmed and T-REx datasets, most of the jointly trained models outperform the GAT baseline that was specifically trained to do link prediction, while on Twitter, our model achieves competitive performance. In experiments comparing the use of graph-based similarity information in training ($\alpha > 0$ vs. $\alpha = 0$) we observe further improvements in performance on Pubmed and T-REx. In some cases, these improvements are quite large: with $\alpha = 0.1$, the masked T-REx model's AUC improves by 0.124, for a 39% improvement in performance over the chance value of 0.5.[4]

Notably, this performance is zero-shot, with no additional training on link prediction. It is also inductive: the joint training set did not include the test-set's graph structure. Finally, because initial node representations were based on the truncated SVD of the adjacency matrix, our node encoders do not utilize text in any way other than via the joint training process with their corresponding text encoders. We believe that this phenomenon—powerful link predictors emerging from a joint graph-text representation learning objective without explicit training—illustrates the power of using text to inform graph-related tasks.

6.2 Language Modeling

On language modeling, we do not observe significant differences in performance between our jointly trained `distilgpt2`-based text encoder and the unimodal LM baselines. Average perplexity is only slightly higher for the joint text encoder than for the fine-tuned unimodal LM: for Pubmed, we record test-set mean perplexity of 6.61 (vs. 6.51 for the baseline); for Twitter, 15.81 (vs. 15.55 baseline); for T-REx, 14.60 (vs. 13.80 baseline). The differences are significant ($p < 0.05$) for Pubmed and Twitter by a bootstrap test over texts, but not for T-REx. In other words, there is at worst a slight reduction in language modeling performance and sometimes no significant change.

6.3 Category Classification

Overall, we see high performance on node category classification relative to the baselines across all three datasets. ConGraT performs on par with or better than the baseline models in 33 out of 36 experiment settings and statistically significantly outperforms baselines in 25 out of 36 experiment settings. In particular, we find that in settings where there is a lower signal in one modality (e.g. tweet text is not as useful in predicting demographic information and graph position is not as useful in predicting a Pubmed category) the incorporation of

[4]Unlike in Subsection 6.3, we do not test for significance, as the proper way to perform bootstrapping on networks is still an open research problem (Levin and Levina, 2021).

		Region		Age		Gender		Occupation	
		Causal	Masked	Causal	Masked	Causal	Masked	Causal	Masked
Text	ConGraT	**45.57**	**47.04**	**30.22**	**32.21**	**61.43**	**67.00**	**87.47**	**78.52**
	Social-LM	33.57	36.31	25.50	26.40	44.51	54.56	48.72	61.31
	LinkBERT	-	41.09	-	30.94†	-	64.77†	-	77.92†
	Unimodal LM	33.47	37.41	25.43	26.57	43.44	54.24	48.72	60.81
	Maj. Class	13.45	13.45	25.43	25.43	37.73	37.73	48.72	48.72
Graph	ConGraT	**50.55**	**54.10**	36.04	**39.16**	**69.55**	**74.03**	**88.22**	**88.32**
	GAT	50.10	50.10	**37.69**†	37.69†	64.92	64.92	78.09	78.09
	Maj. Class	13.36	13.36	25.44	25.44	37.78	37.78	48.79	48.79
Text + Graph	ConGraT	49.85	**51.87**	33.13	35.28	**69.23**	**74.35**	**88.53**	**85.97**
	Social-LM + GAT	**49.89**†	49.39	37.30	**37.37**	66.07	66.21	78.87	78.93
	LinkBERT + GAT	-	49.37	-	**37.37**	-	66.21	-	79.04
	LM + GAT	**49.89**†	49.39	**37.31**	37.36	66.06	66.21	78.87	78.93
	Maj. Class	13.57	13.45	25.36	25.36	37.73	37.73	48.72	48.72

Table 2: Macro F1 scores for predicting various user characteristics on the Twitter dataset. Values with † are not statistically significantly different from ConGraT ($p > 0.05$).

		Pubmed		T-REx	
		Causal	Masked	Causal	Masked
Text	ConGraT	**76.50**	82.25	83.74	82.20
	Social-LM	56.85	80.19	**83.93**†	79.54
	LinkBERT	-	**83.81**	-	**83.35**†
	Unimodal LM	55.51	79.64	82.41	80.88†
	Maj. Class	20.37	20.37	9.90	9.90
Graph	ConGraT	**70.83**	**73.33**	**70.13**	**60.97**
	GAT	42.78	42.78	38.06	38.06
	Maj. Class	20.43	20.43	9.74	9.74
Text + Graph	ConGraT	**78.21**	**82.27**	**84.54**	**81.51**
	Social-LM + GAT	49.65	65.83	41.61	56.00
	LinkBERT + GAT	-	73.15	-	56.05
	LM + GAT	47.83	66.14	41.56	57.16
	Maj. Class	20.37	20.37	9.90	9.90

Table 3: Macro F1 scores for predicting article category for Pubmed and T-REx. Values with † are not statistically significantly different from ConGraT ($p > 0.05$).

the complementary modality can significantly increase performance over a unimodal baseline. One additional observation is that the concatenation of ConGraT text and graph embeddings occasionally performs worse than the corresponding (jointly trained) text or graph embedding on its own. This is possibly due to the simplicity of the concatenation method we used to combine them; more sophisticated fusion methods for combining embeddings could further improve performance. Finally, all presented results were run with 10 bootstrap iterations, and differences with baselines are significant at $p < 0.05$ unless otherwise noted.

Twitter. Macro F1 scores for the category classification tasks on region, age, gender, and occupation are shown in Table 2. When using text embeddings, ConGraT consistently outperforms the various baselines for all categories. Improvements are somewhat smaller when using graph embeddings and they are more mixed when using the concatenations of text and graph embeddings. This is likely because we are predicting node level features that are highly linked with positions in the social follow graph. Thus, as previously demonstrated by other graph-augmented language models (Karpov and Kartashev, 2022; Kulkarni et al., 2021; Yasunaga et al., 2022) incorporating graph information into text is very valuable, while the converse may be slightly less useful. For example, prior work suggests that a substantial share of connections on Twitter lie in the same geographic region (Takhteyev et al., 2012) or in the same age group (Chamberlain et al., 2017).

Pubmed and T-REx. Table 3 shows macro F1 scores for article category classification on Pubmed and T-REx. ConGraT outperforms all of the baselines when using the graph embeddings, while results using the text embeddings are more mixed. Unlike in Twitter, the graph structures of Pubmed and T-REx are more sparse, and each node directly contains the text information which likely contains words related to the topic class. This is likely why we see the largest performance gains when incorporating text information into the graph embeddings rather than incorporating graph information into the text embeddings. Despite this, note that the performance differences in the text setting are relatively small, and ConGraT still outperforms all baselines when using the combination of text and graph embeddings.

7 Conclusion

We propose ConGraT, a contrastive pretraining framework for jointly learning embeddings of graph nodes and texts, applicable to a general class

of problems which feature texts from entities which are connected by a graph structure. Because our method is self-supervised, pretraining requires no labeled or other downstream data, and the learned representations are not specific to any particular downstream task. The models are also inductive, able to generalize to entirely new graphs rather than only new texts from the training graph. In experiments on social, citation, and link graphs, our method outperforms a number of baselines in link prediction and node category classification, without meaningful degradation in language modeling performance. These results suggest opportunities to build on this work, from analytical uses (e.g., detection or interpretation of network communities) to methodological improvements such as exploring other contrastive objectives.

8 Ethical Considerations

Turning to the ethical implications of our work, applying this method to existing social or other networks poses the risk of learning and reproducing biases or toxic behavior exhibited in those datasets (Liang et al., 2021). The risk of learning such harmful behavior is likely to be greatest on social media datasets, given the greater prevalence of harassment and toxic or "antisocial" behavior there (Saveski et al., 2021; Atske, 2021). On the other hand, for applications like detecting hate speech and toxicity, this may be the intended behavior; careful attention to the ethics of how any model is used is key.

On the dataset side, we believe there are no meaningful concerns about release of personally identifying information (PII) or offensive content with the Pubmed and T-REx datasets; both are already public and vetted by respectively journals and the Wikipedia community. We do not believe that such concerns prevent the release of our Twitter data either: the tweets themselves are public and the users included are public figures.

9 Limitations

Like all models, ours has certain limitations. Most fundamentally, we make the assumption that there is a relationship between the graph structure over nodes and the texts associated with each node. If the two modalities are generated independently, or otherwise do not convey any useful information about each other, joint pretraining should not be expected to improve performance.

A more practical limitation is around the scale of graph data. Current graph neural networks require large amounts of memory, and scaling to very large graphs (Serafini and Guan, 2021; Ma et al., 2022) is an active research area. It is accordingly unclear how to apply our approach to very large graphs. Because our model is inductive, training repeatedly on (the subgraphs over) random samples of nodes may work, but we have not evaluated this approach.

A further limitation of the present paper's analysis is that we have not validated a directed analogue of similarity to be used with positive values of α in directed graphs. While one has the option of either using $\alpha = 0$ or discarding edge directions, we aim to remedy this issue in future work.

Acknowledgements

The authors gratefully acknowledge support in the form of access to data from Twitter, Inc. We are also indebted to colleagues at the MIT Center for Constructive Communication for their feedback on earlier versions of this work.

References

Sara Atske. 2021. The State of Online Harassment. Technical report, Pew Research Center.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. ArXiv:1607.06450 [cs, stat].

Benjamin Paul Chamberlain, Clive Humby, and Marc Peter Deisenroth. 2017. Probabilistic inference of twitter users' age based on what they follow. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 191–203. Springer.

Shantanu Chandra, Pushkar Mishra, Helen Yannakoudakis, Madhav Nimishakavi, Marzieh Saeidi, and Ekaterina Shutova. 2020. Graph-based Modeling of Online Communities for Fake News Detection. ArXiv:2008.06274 [cs].

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Munmun De Choudhury, Hari Sundaram, Ajita John, Doree Duncan Seligmann, and Aisling Kelliher. 2010. "Birds of a Feather": Does User Homophily Impact Information Diffusion in Social Media? ArXiv:1006.1702 [physics].

Benjamin Elizalde, Shuayb Zarar, and Bhiksha Raj. 2019. Cross Modal Audio Search and Retrieval with Joint Embeddings Based on Text and Audio. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4095–4099, Brighton, United Kingdom. IEEE.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

William Falcon, Jirka Borovec, Adrian Wälchli, Nic Eggert, Justus Schock, Jeremy Jordan, Nicki Skafte, Ir1dXD, Vadim Bereznyuk, Ethan Harris, Tullie Murrell, Peter Yu, Sebastian Præsius, Travis Addair, Jacob Zhong, Dmitry Lipin, So Uchida, Shreyas Bapat, Hendrik Schröter, Boris Dayma, Alexey Karnachev, Akshay Kulkarni, Shunta Komatsu, Martin.B, Jean-Baptiste SCHIRATTI, Hadrien Mary, Donal Byrne, Cristobal Eyzaguirre, Cinjon, and Anton Bakhtin. 2020. PyTorchLightning/pytorch-lightning: 0.7.6 release.

Matthias Fey and Jan Eric Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *Proceedings of the ICLR Workshop on Representation Learning on Graphs and Manifolds*.

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. 1998. CiteSeer: an automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries - DL '98*, pages 89–98, Pittsburgh, Pennsylvania, United States. ACM Press.

Antoine Gourru, Julien Velcin, and Julien Jacques. 2020. Gaussian Embedding of Linked Documents from a Pretrained Semantic Space. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3912–3918, Yokohama, Japan. International Joint Conferences on Artificial Intelligence Organization.

Shashank Gupta and Vasudeva Varma. 2017. Scientific Article Recommendation by using Distributed Representations of Text and Graph. In *Proceedings of the 26th International Conference on World Wide Web Companion - WWW '17 Companion*, pages 1267–1268, Perth, Australia. ACM Press.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Dan Hendrycks and Kevin Gimpel. 2020. Gaussian Error Linear Units (GELUs). ArXiv:1606.08415 [cs].

Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '02*, page 538, Edmonton, Alberta, Canada. ACM Press.

Ilia Karpov and Nick Kartashev. 2022. SocialBERT – Transformers for Online Social Network Language Modelling. In Evgeny Burnaev, Dmitry I. Ignatov, Sergei Ivanov, Michael Khachay, Olessia Koltsova, Andrei Kutuzov, Sergei O. Kuznetsov, Natalia Loukachevitch, Amedeo Napoli, Alexander Panchenko, Panos M. Pardalos, Jari Saramäki, Andrey V. Savchenko, Evgenii Tsymbalov, and Elena Tutubalina, editors, *Analysis of Images, Social Networks and Texts*, volume 13217, pages 56–70. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. In *Bayesian Deep Learning Workshop (NeurIPS 2016)*.

Vivek Kulkarni, Shubhanshu Mishra, and Aria Haghighi. 2021. LMSOC: An approach for socially sensitive pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2967–2975, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Keith Levin and Elizaveta Levina. 2021. Bootstrapping Networks with Latent Space Structure. ArXiv:1907.10821 [math, stat].

Chang Li and Dan Goldwasser. 2019. Encoding Social Information with Graph Convolutional Networks forPolitical Perspective Detection in News Media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2594–2604, Florence, Italy. Association for Computational Linguistics.

Chang Li, Yi-Yu Lai, Jennifer Neville, and Dan Goldwasser. 2017. Joint Embedding Models for Textual and Social Analysis. In *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia.

Jinning Li, Shubhanshu Mishra, Ahmed El-Kishky, Sneha Mehta, and Vivek Kulkarni. 2022. NTULM: Enriching Social Media Text Representations with Non-Textual Units. In *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Yangguang Li, Feng Liang, Lichen Zhao, Yufeng Cui, Wanli Ouyang, Jing Shao, Fengwei Yu, and Junjie Yan. 2021. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. In *International Conference on Learning Representations*.

Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2021. Towards Understanding and Mitigating Social Biases in Language Models. In *Proceedings of the 2021 International Conference on Machine Learning*, pages 6565–6576.

Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. 2018. Content to Node: Self-Translation Network Embedding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1794–1802, London United Kingdom. ACM.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *Proceedings of the Seventh International Conference on Learning Representations (ICLR 2019)*, New Orleans, LA, USA.

Hehuan Ma, Yu Rong, and Junzhou Huang. 2022. Graph Neural Networks: Scalability. In Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao, editors, *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 99–119. Springer Nature Singapore, Singapore.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval*, 3(2):127–163.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed Precision Training. In *Proceedings of the 2018 International Conference on Learning Representations*.

Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. 2022. Slip: Self-supervision meets language-image pre-training. In *European Conference on Computer Vision*, pages 529–544. Springer.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation Learning with Contrastive Predictive Coding. ArXiv:1807.03748 [cs, stat].

Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, and Georg Rehm. 2022. Neighborhood contrastive learning for scientific document representations with citation embeddings. *arXiv preprint arXiv:2202.06671*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Advances in Neural Information Processing Systems 32*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. Curran Associates, Inc.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages

3980–3990, Hong Kong, China. Association for Computational Linguistics.

Gerard Salton. 1988. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley series in computer science. Addison-Wesley, Reading, Mass.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*. Curran Associates, Inc.

Martin Saveski, Brandon Roy, and Deb Roy. 2021. The Structure of Toxic Conversations on Twitter. In *Proceedings of the Web Conference 2021*, pages 1086–1097, Ljubljana Slovenia. ACM.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazine*, 29(3):93.

Marco Serafini and Hui Guan. 2021. Scalable Graph Neural Network Training: The Case for Sampling. *ACM SIGOPS Operating Systems Review*, 55(1):68–76.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867. Curran Associates, Inc.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Yuri Takhteyev, Anatoliy Gruzd, and Barry Wellman. 2012. Geography of twitter networks. *Social Networks*, 34(1):73–81. Capturing Context: Integrating Spatial and Social Network Analyses.

Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-Aware Network Embedding for Relation Modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1722–1731, Vancouver, Canada. Association for Computational Linguistics.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR)*.

Isaac Waller and Ashton Anderson. 2021. Quantifying social organization and political polarization in online platforms. *Nature*, 600(7888):264–268. Number: 7888 Publisher: Nature Publishing Group.

Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network representation learning with rich text information. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 2111–2117, Buenos Aires, Argentina. AAAI Press.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022. LinkBERT: Pretraining Language Models with Document Links. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8003–8016, Dublin, Ireland. Association for Computational Linguistics.

Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2017. User Profile Preserving Social Network Embedding. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3378–3384, Melbourne, Australia. International Joint Conferences on Artificial Intelligence Organization.

Haopeng Zhang and Jiawei Zhang. 2020. Text Graph Transformer for Document Classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8322–8327, Online. Association for Computational Linguistics.

A Dataset Details

Here we discuss the datasets in more detail and provide summary statistics in Table 4 and Table 5 on the target variables.

Twitter. To obtain the age and gender of Twitter users, we connected the accounts to their corresponding Wikipedia pages and used Wikidata to infer those two features. Users also self-report locations in their Twitter bios; from these locations, we created four regional categories to predict. Finally, we used data from Ballotpedia[5] to label whether a user is a politician or not.

T-REx. We used the articles in the T-REx corpus (Elsahar et al., 2018) of Wikipedia articles that were labeled with the "Robots" category or any of its descendants. From these categories, we constructed several binary target label sets for the T-REx prediction task. However, since the most commonly occurring category was only associated with 526 (roughly 5.7%) of the articles, we expanded each article's labels to include both first and second level ancestors in the category hierarchy to obtain better class label balance. From the initial set of 1,433 unique categories, this expansion yielded a total of 6,643 unique categories, with the most frequent

[5] https://ballotpedia.org/.

Feature	Category	# Nodes
Region	Midwest	24
	Northeast	250
	South	319
	West	267
Age	19-39	108
	40-49	444
	>=65	168
Gender	Female	323
	Male	491
Occupation	Non-politician	1653
	Politician	85

Table 4: Breakdown of various demographic features of Twitter users.

Dataset	Article Category	# Nodes
Pubmed	Experimental	375
	Type I	881
	Type II	740
T-REx	Robots	37
	Rockets	112
	Sci-Fi	72
	Spacecraft	138
	Space Telescopes	57

Table 5: Breakdown across article categories for Pubmed and T-Rex data.

("Spacecraft") occurring on 1,342 articles. We then selected five categories to use as labels for separate binary prediction tasks, choosing frequent categories that generally had small overlap with each other (i.e. were associated with mostly disjoint document sets.) The resultant categories we selected are listed in Table 5.

B Model Architectures and Training Details

We estimate that training all of our joint and baseline models together used 192 hours of GPU time. Because the assumptions made for this value are conservative, the actual value is likely slightly less.

B.1 Joint Models

We trained all ConGraT models on either a single NVIDIA RTX A6000 GPU or a single NVIDIA A100 GPU. For masked LM experiments, we used the pretrained `all-mpnet-base-v2` model (Song et al., 2020) from the sentence-transformers toolkit (Reimers and Gurevych, 2019), which has 12 layers of 12 heads each, producing 768-dimensional embeddings. It was pretrained constrastively on several corpora from similar domains to those we consider here,[6] making it a good match for our work. Our causal LM experiments used the pretrained `distilgpt2` model (Sanh et al., 2019), distilled from the larger GPT-2 model (Radford et al., 2019), with 6 layers of 12 heads each, producing 768-dimensional embeddings.[7] For the graph node encoder, all models used a graph attention network (GAT) (Veličković et al., 2018) with 3 layers and 2 attention heads in each layer. As in a standard transformer, each graph convolutional layer is separated from the next by a linear layer, with layer normalization (Ba et al., 2016) applied afterwards. Hidden representations are 64-dimensional, and the final output vectors are 768-dimensional so that baseline model outputs have the same shape as language model outputs.

Parameter counts are as follows: `distilgpt2`, 81.9 million; `all-mpnet-base-v2`, 109.4 million; our GAT encoder, 199.7 thousand. The jointly trained models, including the adapter layers after the text and graph encoders, have 83.9 million parameters (causal / `distilgpt2`) and 110.9 million parameters (masked / `all-mpnet-base-v2`).

Training is quite sensitive to the learning rate; we found that a good compromise between speed of training and instability was a value of `1e-4`. At a variety of learning rates, there were also intermittent large spikes in the norm of the gradient, which derailed training unless the gradients were clipped. We clipped the gradient at each step to a norm of 1. In order to reduce memory consumption and fit larger batches onto a single GPU, we used 16-bit mixed precision training (Micikevicius et al., 2018). We encountered numerical overflow problems with FP16, however, related to large gradient values at certain layers, and found it necesary to reduce the init-scale parameter of the gradient scaler from its default value of 2^{16} to 256 in order to avoid overflow. We initialized the log-temperature parameter τ to 3.5 and constrained it to be within $(-\log 100, +\log 100)$ in order to avoid training instability. We trained all models with PyTorch (Paszke et al., 2019) and pytorch-lightning (Falcon et al., 2020), also using pytorch-geometric (Fey and Lenssen, 2019) for graph encoders and GAT baselines, and Huggingface Transformers (Raffel et al., 2020) for textual baselines and text encoders.

We also found that performance suffers if each

[6] See the model card for details: https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

[7] Again see the model card for more details: https://huggingface.co/distilgpt2.

batch is not unique on nodes (i.e., if each node has multiple texts, only one text per node can be in any given batch). We experimented with simply dropping duplicates from uniformly sampled batches, but this discarded too much data. Instead, we randomly sorted the texts on each epoch so as to minimize the number of within-batch duplicates (assuming minibatches are taken consecutively from the sorted dataset), and dropped any remaining duplicates.

Finally, because the objective is batch-wise contrastive, the problem becomes quadratically more difficult as the batch size increases. We used the largest batch size we could consistently fit into available hardware, but future work should explore the question of returns to scale.

All models used the AdamW optimizer (Loshchilov and Hutter, 2019) with β values of (0.9, 0.999) and without weight decay. All joint models used a probability of 0.3 for dropout applied to text and node embeddings. Learning rates and batch sizes for our various models are shown in Table 6.

B.2 Unimodal Baselines

Our unimodal baselines were trained on up to four NVIDIA GTX 1080 Ti GPUs. To better understand the effects of multi-modal pretraining, we also trained unimodal models, either language models or graph attention transformers, and evaluated these unimodal models on the downstream tasks. For textual models, we fine-tuned pretrained `all-mpnet-base-v2` and `distilgpt2` on the downstream datasets. Language models were fine-tuned for 3 epochs. For graph models, we trained graph attention network (GAT) models to do non-variational graph autoencoding (Kipf and Welling, 2016), also known as link prediction, on the network structure of the downstream-task datasets. GAT models were trained from between 30 to 300 epochs with early stopping based on validation AUC, with patience of 5 epochs and minimum delta of 0.001. We compare these unimodal baselines against ConGraT. Parameter counts for the text and graph baselines are the same as reported for the appropriate modality's joint encoder in Subsection B.1. Batch sizes and learning rates, as for joint models, are reported in Table 6.

B.3 Social-LM

We implemented a baseline Social-LM, as a modified version of SocialBERT[8] (Karpov and Kartashev, 2022) (also very closely related to LMSOC (Kulkarni et al., 2021)), which uses pretrained, frozen node embeddings to prime language model pretraining. Specifically, we added a special node token [G] at the beginning of texts and used the pretrained GAT model to obtain the corresponding node embedding paired with each tweet or article, which was used to replace the token embedding for [G]. During the language model pretraining, we froze the node embeddings and only fine-tuned the language model to generate texts conditioned on the node embeddings. Our Social-LM implementation has some key differences from SocialBERT and LMSOC: (1) for masked LM experiments, we used `all-mpnet-base-v2` to replace BERT, to be consistent with other experiments for a fair comparison; (2) we also experimented with a causal language model `distilgpt2` under the Social-LM baseline, whereas LMSOC and SocialBERT only used the masked language model BERT; (3) we injected the node embedding as the zero token embedding of texts as SocialBERT suggests, whereas LMSOC appends the node embedding at the end. We adopted the zero token injection approach because the same strategy is adaptable for both causal and masked language modeling, while last token injection does not work for causal LMs like `distilgpt2`; (4) we used our unimodal GAT model trained on the graph autoencoding task to generate node embeddings for each tweet or article, whereas LMSOC uses node2vec and SocialBERT uses vectors from SVD and Deep Walk. We used the GAT in order to be consistent with ConGraT and the unimodal baseline, to ensure that the comparisons were fair, and because it was likely to be a stronger baseline than using SVD. Social-LM models were fine-tuned for 3 epochs with the same hyperparameters used for the language modeling baseline, and have the same number of parameters as `all-mpnet-base-v2`, our masked LM baselines and the joint masked text encoders.

B.4 LinkBERT

We implemented and trained LinkBERT (Yasunaga et al., 2022) as described in the original paper, with the only difference being that we used the same `all-mpnet-base-v2` architecture

[8]The SocialBERT paper did not publish their code.

Model or Model Family	Batch Size	Base LR	LR Schedule
ConGraT	36	1.0e-4	Constant LR
LM Baseline	12	5.0e-5	Linear 10% warmup + 90% decay
SocialLM	12	5.0e-5	Linear 10% warmup + 90% decay
LinkBERT	12	5.0e-5	Linear 10% warmup + 90% decay
GNN Baseline, T-REx, Dir.	n/a	1.0e-3	Constant LR
GNN Baseline, T-REx, Undir.	n/a	1.0e-3	Constant LR
GNN Baseline, Pubmed, Dir.	n/a	1.0e-2	Constant LR
GNN Baseline, Pubmed, Undir.	n/a	1.0e-3	Constant LR
GNN Baseline, Twitter, Dir.	n/a	1.0e-2	Constant LR
GNN Baseline, Twitter, Undir.	n/a	1.0e-2	Constant LR

Table 6: Batch sizes and learning rates for all models. Each model shown used the provided batch size and learning rate for all datasets. All joint models, whether directed or undirected, with $\alpha = 0$ or $\alpha = 0.1$, and causal or masked encoders, use same batch size and learning rate. Except for the GNN baseline learning rate, where we tried both `1.0e-2` and `1.0e-3` and found large dataset-specific effects on performance, all models listed in the same model family use the same parameter settings. GNN baselines do not list a batch size because the entire graph is processed at once.

as the other baseline models (instead of BERT-Base) in order to maintain consistency across experiments. We initialized weights from the pre-trained `all-mpnet-base-v2` model from sentence-transformers, and fine-tuned it on the masked language modeling (MLM) and document relation prediction (DRP) tasks for 3 epochs. Hyperparameters used for training are listed in Table 6. Note that because of its MLM training objective, we used LinkBERT as a baseline for masked language model variants of ConGraT only. All LinkBERT models have the same number of parameters as `all-mpnet-base-v2`, as the DRP head is dropped at inference time.

We created training instances for LinkBERT by sampling contiguous, linked, or random text segment pairs for the DRP training objective from each dataset, with the three options appearing uniformly (33%, 33%, 33%). For the Pubmed and Twitter datasets, we sampled 100,000 text pairs for each category, for a total of 300,000 pairs. For T-REx, which is a substantially smaller dataset, we sampled 10,000 text pairs for each category, for a total of 30,000 pairs. Text pairs consisted of anchor text segment X_A and paired text segment X_B: (X_A, X_B). The specific methods we used for sampling pairs for each dataset were as follows:

Pubmed. Text segments in each pair consisted of individual sentences from the abstracts of each article in the dataset. Anchor segments X_A were taken by sampling a random abstract, then sampling a random sentence from that abstract. For continuous pairs, X_B was chosen as the sentence immediately following X_A in the abstract (X_A could not be the last sentence of the abstract). For linked pairs, X_B was chosen as a random sentence from the abstract of one of the articles that was connected to X_A's corresponding article in the citation graph. For random pairs, X_B was chosen as a random sentence from an abstract whose article was not connected to X_A's corresponding article in the citation graph.

T-REx. Text segments in each pair consisted of individual sentences from the introductory paragraphs of each article in the dataset. Anchor segments X_A were taken by sampling a random article, then sampling a random sentence from that article's introductory paragraphs. For continuous pairs, X_B was chosen as the sentence immediately following X_A, with the same restriction as in Pubmed that X_A could not be the last sentence. For linked pairs, X_B was chosen as a random sentence from the introductory paragraphs of one of the articles that was connected to X_A's corresponding article in the link graph. For random pairs, X_B was chosen as a random sentence from an article that was not connected to X_A's corresponding article in the link graph.

Twitter. Twitter has a different graph-text structure compared to Pubmed and T-REx; rather than the nodes consisting of texts themselves, the nodes are users who can each produce multiple tweets. Therefore, the notion of what constitutes continuous or linked text segments (tweets) is less clearly defined. We defined these relationships as follows. For continuous pairs, we sampled a random tweet as X_A, and chose X_B as a different tweet from the same user as X_A. For linked pairs, we sampled

X_A from the set of tweets that *mentioned* other users that were present in our dataset. Then, X_B was chosen as a random tweet from the mentioned user. Random pairs were simply taken by randomly sampling two tweets from different users to use as X_A and X_B.

B.5 Downstream Logistic Regression

We use the standard *scikit-learn* logistic regression implementation with the default *lbfgs* solver and L2 regularization and increase the maximum iterations for the solver to 10000.

C Summary Evaluation Metrics

In addition to downstream applications like node classification and link prediction, we also examine certain summary metrics of the quality of our joint models, without regard to any particular downstream task. The metrics are intended to measure how well the models integrate information across language and graph modalities. We examine specifically:

- Top-k accuracy in selecting the node which produced a given text ($k = 1, ..., 10$). For each text, we select the user with highest cosine similarity between the user's node embedding and the text's embedding. Note that in retrieval contexts, this might itself be an important downstream measure.
- Embedding distance correlation: the correlation of the cosine similarities between pairs of texts with the cosine similarities between the corresponding pairs of nodes.
- Correlation of text embedding distance with a purely graph-based distance (we use SimRank), extending the previous metric by grounding text embedding distance more directly in the graph.

We compare each joint model's results on these metrics to those achieved by the combination of a language model and non-variational graph autoencoder, or link predictor (Kipf and Welling, 2016), each with the same initialization and architecture as the joint models' text and node encoders, trained separately on the same data. Joint models initialized with causal language models as text encoders are compared to causal models fine-tuned causally, and analogously for models with masked text encoders.

C.1 Results

We observe a meaningful increase in several measures of cross-modal integration. As shown in Figure 3, top-k accuracy is substantially higher than the separately-trained baseline for all models at all values of k. All differences are significant at the $\alpha = 10^{-6}$ level according to a bootstrap test. Moreover, the top-k accuracies achieved are often high relative to the size of the datasets. With 1,996 articles in the Pubmed test set, the best-performing model includes the correct article for a text snippet in its top 10 most similar articles (0.5% of the test set) 26.6% of the time.

Results are similar for other metrics, with graph and text embedding distances significantly more correlated after training than before for all models against their respective baselines ($p < 10^{-6}$). In 16 out of 18 cases, we also observe a significant increase ($p < 10^{-6}$) in the correlation between the cosine similarity of text embeddings and the SimRank similarity of the corresponding nodes.

We also observe certain patterns in these metrics by type of model. First, using $\alpha = 0.1$ usually but not always leads to higher scores compared to the corresponding model with $\alpha = 0$, including on top-k accuracy. This fact illustrates both the potential value of incorporating this information into the loss and also the need to tune the hyperparameter in practice. We do not, however, see models with either masked or causal text encoders consistently outperform models with the other kind of text encoder.[9] Finally, we note that for every dataset and choice of masked/causal text encoder, the best-performing model on these metrics is one which discards edge directions.

D Downstream Task Accuracy

We recreate Table 2 and Table 3 with accuracy instead of F1 scores. The results are shown in Table 8 and Table 9.

E Terms of Use

We used certain scientific artifacts in preparing this paper, specifically pretrained models, software and datasets. Software we used was all available under open-source licenses which permit our use for research purposes. Pretrained models were also

[9] But note that this comparison is confounded by the difference in size between the masked and causal encoders we use; MPNet has about 109 million parameters while distilgpt2 has only 82 million.

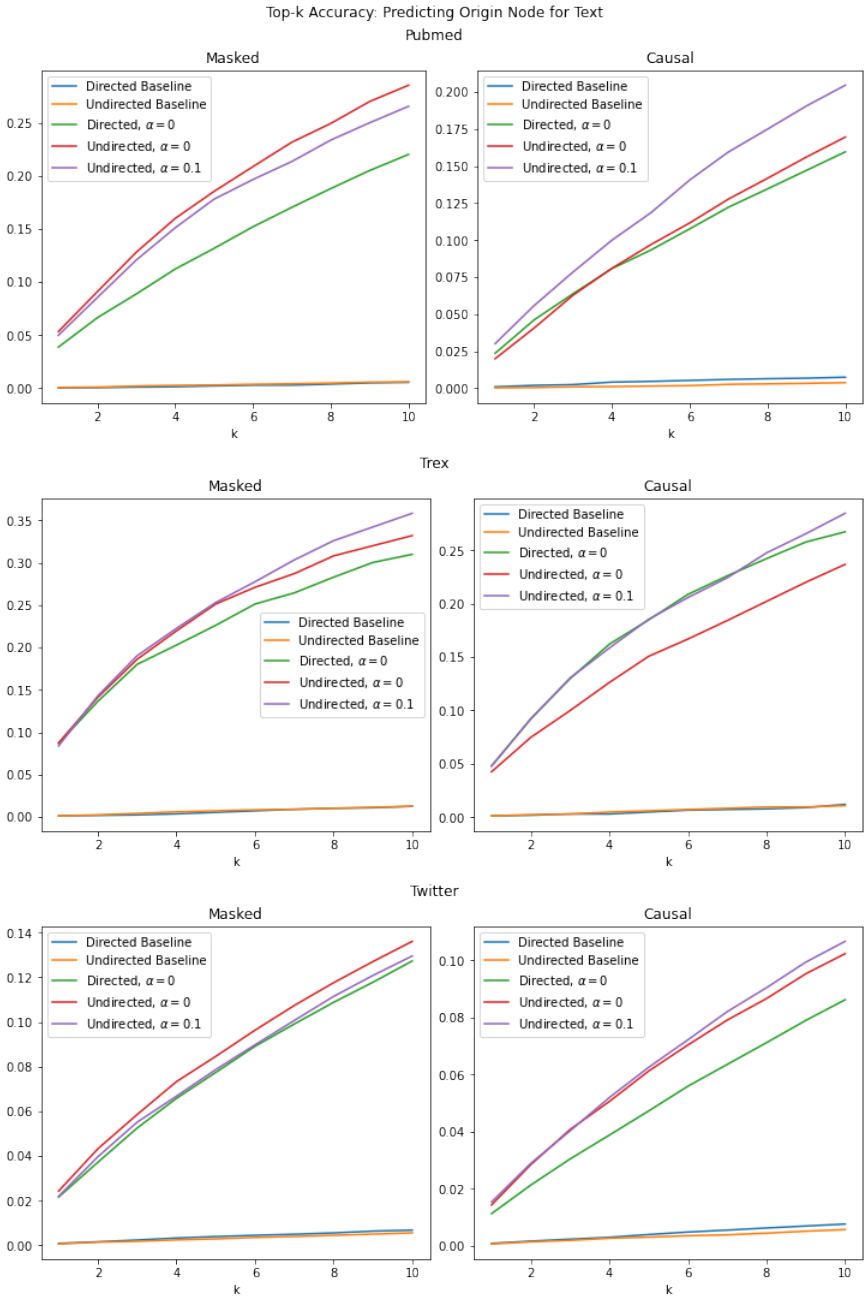

Figure 3: Top-k accuracy at selection of the node which produced a text, for various values of k. "Baseline" indicates the use of separately pretrained embeddings, and other results are for models with various combinations of edge-direction use and graph-similarity information. Models with $\alpha = 0$ used no such information in their training losses, while models with $\alpha = 0.1$ put 10% weight on that information and 90% on the correct node/text correspondence.

Dataset	Directed	LM Type	Sim.	Distance Coupling		Emb. vs Graph: Text	
				Joint	Baseline	Joint	Baseline
Pubmed	Directed	Causal	$\alpha = 0.0$	**0.486**	-0.009	**0.072**	0.018
		Masked	$\alpha = 0.0$	**0.373**	0.019	**0.097**	0.064
	Undirected	Causal	$\alpha = 0.0$	**0.505**	-0.000	**0.116**	0.026
			$\alpha = 0.1$	**0.543**	-0.000	**0.126**	0.026
		Masked	$\alpha = 0.0$	**0.323**	0.019	**0.171**	0.086
			$\alpha = 0.1$	**0.382**	0.019	**0.167**	0.086
T-REx	Directed	Causal	$\alpha = 0.0$	**0.358**	0.041	**0.071**	0.018
		Masked	$\alpha = 0.0$	**0.155**	0.013	**0.160**	0.083
	Undirected	Causal	$\alpha = 0.0$	**0.499**	0.028	**0.252**	0.021
			$\alpha = 0.1$	**0.579**	0.028	**0.273**	0.021
		Masked	$\alpha = 0.0$	**0.286**	0.058	**0.267**	0.151
			$\alpha = 0.1$	**0.356**	0.058	**0.302**	0.151
Twitter	Directed	Causal	$\alpha = 0.0$	**0.305**	0.029	**0.047**	0.022
		Masked	$\alpha = 0.0$	**0.265**	0.070	**0.052**	† 0.050
	Undirected	Causal	$\alpha = 0.0$	**0.331**	0.036	**0.039**	0.022
			$\alpha = 0.1$	**0.310**	0.036	**0.050**	0.022
		Masked	$\alpha = 0.0$	**0.292**	0.076	0.046	† **0.046**
			$\alpha = 0.1$	**0.257**	0.076	**0.056**	0.046

Table 7: Results of all six trained models for each dataset on two further metrics, beyond the top-k accuracy of Figure 3, measuring cross-modal integration of graph and text information. Baselines are separately pretrained models with the same edge-direction status and type of language model (causal, with `distilgpt2`, or masked, with `all-mpnet-base-v2`). The best-performing model in each experiment, joint or baseline, is shown in **bold**. Note that all joint-vs-baseline differences are statistically significant at the $\alpha = 10^{-6}$ level under percentile-bootstrap tests, except for the two correlations of text-embedding distance with graph distance marked with daggers. Metrics are described in more detail in Subsection 5.3.

available under open-source licenses permitting research use, such as the `distilgpt2` model we obtained from the HuggingFace Hub which is subject to the Apache 2.0 license. Finally, two of our datasets (Pubmed and T-REx) were available under Creative Commons or other permissive open-source licenses. We have access to Twitter data pursuant to a special-purpose agreement with Twitter permitting use of data for research and publication.

Our code and datasets, when released upon publication, will be subject to an open-source license allowing use for research purposes.

		Region		Age		Gender		Occupation	
		Causal	Masked	Causal	Masked	Causal	Masked	Causal	Masked
Text	ConGraT	**62.01**	**63.98**	**61.73**	62.17	**68.39**	**71.75**	**97.73**	96.65
	Social-LM	52.64	54.84	61.64†	61.28†	61.33	66.22	95.00	95.53
	LinkBERT	-	58.69	-	**62.37**†	-	71.08†	-	**96.71**†
	Unimodal LM	52.54	55.66	61.67†	61.48†	61.16	65.85	95.00	95.51
	Maj. Class	36.79	36.79	61.67	61.67	60.61	60.61	95.00	95.00
Graph	ConGraT	**67.79**	**72.21**	**63.19**	**63.69**	**72.51**	**76.44**	**97.89**	**97.97**
	GAT	67.05†	67.05	63.00†	63.00†	69.07	69.07	96.89	96.89
	Maj. Class	36.56	36.56	61.72	61.72	60.74	60.74	95.28	95.28
Text + Graph	ConGraT	**66.82**	**69.37**	61.56	**62.37**	**72.68**	**76.90**	**97.87**	**97.54**
	Social-LM + GAT	66.72†	66.04	**62.11**†	62.15†	69.92	70.04	96.78	96.78
	LinkBERT + GAT	-	66.02	-	62.14†	-	70.06	-	96.79
	LM + GAT	66.72†	66.04	**62.11**†	62.15†	69.92	70.04	96.78	96.78
	Maj. Class	37.26	36.79	61.43	61.43	60.61	60.61	95.00	95.00

Table 8: Accuracies for predicting various user characteristics on the Twitter dataset using both graph and text embeddings (Text + Graph) or these embeddings by themselves. Models with a dagger (†) are not statistically significantly significant from ConGraT ($p > .05$).

		Pubmed		TRex	
		Causal	Masked	Causal	Masked
Text	ConGraT	**77.34**	83.56	84.55	85.15
	Social-LM	59.13	80.06*	**87.40**	85.84*
	LinkBERT	-	**84.22**	-	**88.97**
	Unimodal LM	58.27	79.46*	87.29	86.13
	Maj. Class	44.00	44.00	32.00	32.21
Graph	ConGraT	**73.86**	**76.29**	**72.98**	**67.64**
	GAT	50.60	50.60	43.75	43.75
	Maj. Class	44.19	44.19	32.21	32.21
Text + Graph	ConGraT	**79.38**	**83.98**	**85.61**	**84.43**
	Social-LM	53.73	66.96	45.98	58.30
	LinkBERT	-	74.47	-	58.20
	LM + GAT	53.10	67.06	45.94	59.26
	Maj. Class	44.00	44.00	34.19	32.93

Table 9: Accuracy for predicting article category for Pubmed and T-REx datasets. Values with a dagger (†) are not statistically significantly different from ConGraT ($p > 0.05$).