Label-Metadata guided Document Classification via Graph Transformer

Parth Shroff
pms2384
pshroff@cs.utexas.edu
Department of Computer Science
Department of Electrical & Computer Engineering

Zengqi Liu
zl9594
zengqi-liu@utexas.edu
Department of Computer Science

Chitrank Gupta
cg46773
chitrank@utexas.edu
Department of Computer Science

Abstract

Text classification models have primarily been using input-side features like input text or input images as input to supervised learning models. However, given the increase in the size of document collections paired with the increased number of labels required to accurately classify documents, new approaches are required to effectively scale for text classification. Thus, state-of-the-art models utilize the semantic relationship between documents and the label hierarchy. Our project explored this label hierarchy from two datasets: Web of Science (WOS) Core Collection and the MultiEURLEX (Eurlex) dataset. WOS contains a simple 2-level hierarchy and Eurlex contains a label directed-acyclic-graph (DAG). Doing classification at every internal node of the label hierarchy, as done in past works such as [1] can be inefficient and slow on large datasets, and these approaches have a suboptimal use of the DAG label hierarchy for text classification. We propose a Graph Transformer model that addresses these challenges by utilizing the DAG-based label hierarchy for text classification. The GNN approach also unifies both the label-text and label-hierarchy approaches, whereas many prior approaches only consider one or the other (HDLTex[1] and HiMatch[2] respectively). We also release our code[1].

1 Introduction

We aim to combine label text and label hierarchies when performing document classification. There are some important precursors in the XML space that inspired our model. The first is the Hierarchical Deep Learning for Text Classification (HDLTex)[1]. Typical classifiers may only classify documents into high-level fields like Computer Science or Medical Science but cannot scale for specialized classification of Computer Science or Medial Science fields, given the exponential increase in labels.

[1] https://github.com/theartpiece/LabelMetadata

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

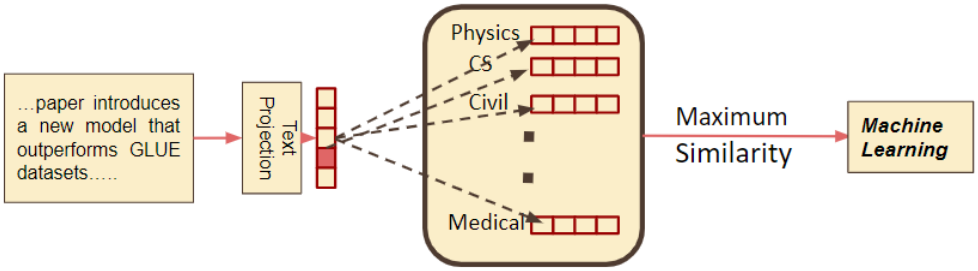

Figure 1: Typical document classification model using direct label text encodings

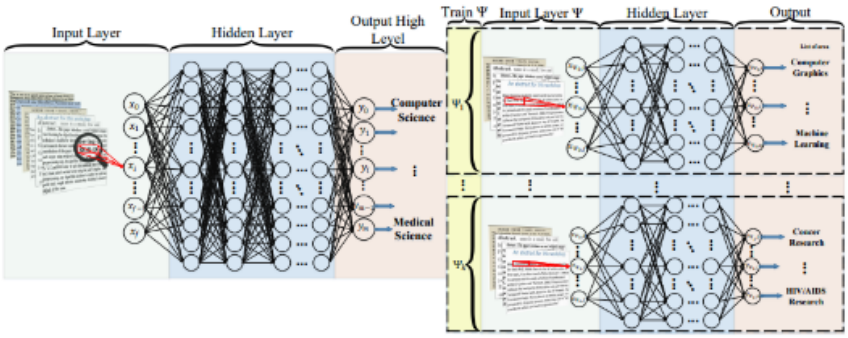

Figure 2: Figure from HDLTex [1] demonstrating the classification architecture with DNN's at each level

HDLTex aims to classify documents into specialized fields and specific sub-labels in addition to efficient classification of high-level labels. HDLTex introduces classifiers into every coarser label of the hierarchy to predict finer labels. Then, during the inference stage, it selects the label with the highest score at every level of the hierarchy and returns the leaf node with the highest score as the predicted label.

Another important approach to hierarchical text classification is the Hierarchy-aware Label Semantics Matching Network (HiMatch)[2] model. In this model, the goal is to align the text semantics and label semantics into a joint embedding space. The joint text-label embedding is created by separate text and label encoders that extract the semantic information of the text and fine-grained labels, respectively. Once the embeddings are extracted, they are projected into a joint text-label embedding space where the text embedding is aligned with the label representation. The goal is to have text that matches certain labels live closer in the text-label embedding space and text with incorrect labels reside farther from each other in the joint-embedding space. Thus, the loss function is also applied to the joint embedding space (using a joint-embedding loss).

The first model type for document classification is the type that utilizes the label text. As shown in Figure 1, the model takes in a BERT encoding of the document and contains a BERT encoding of the pool of labels. After training, the model will find the maximum similarity between the document and the target label in the embedding space and return the appropriate label.

Figure 2 depicts the HDLTex model that aims to utilize the WOS 2-level hierarchy for document classification. There is a deep neural network at the first level that classifies the document into appropriate level 1 labels. Then, based on the prediction from the first layer, a separate DNN is trained to predict the appropriate level 2 labels.

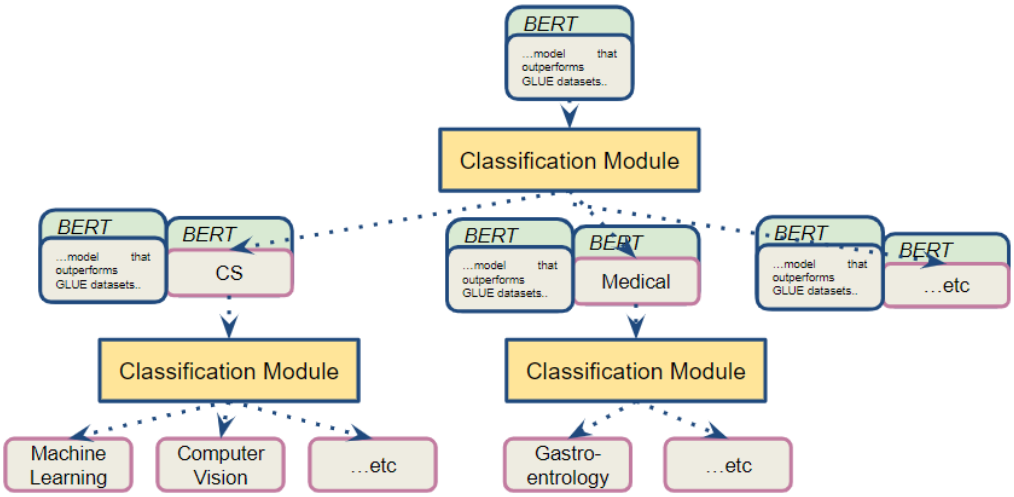

Figure 3: Architecture of our level-wise classification with label-text encoding

Our approach unifies both information via a graph transformer that operates on both the label text embedding and the DAG-based label hierarchy.

2 Methodology

2.1 Flat Classification

The Flat Classification model is a simple dense neural network consisting of several linear layers intermitted by GELU non-linear activations. In this model, we directly classify into the label set and train with a suitable loss function: CrossEntropy loss in the WOS dataset and Binary-CrossEntropy Loss for the Eurlex dataset since Eurlex is a multilabel setting.

2.2 Level-wise Classification

As pointed out earlier, the problem with HDLTex[**hdltex**] is that it's very parameter inefficient as it uses a different deep neural network for classification at every internal node of the label tree. To solve this issue, we use the Large Language Model- sentence-transformer [3] to help improve the model's generalization capabilities without training many parameters. As illustrated in Fig 3 the idea is to use the same text encoder to encode the labels and concatenate them with the document encoding before doing the classification at the fine-grain level.

2.3 Graph Transformer Model

The intuitiveness of the level-wise classification model comes at the cost that it's difficult to implement when the label graph is complicated– for example, the label graph is a directed-acyclic graph (DAG), i.e., it contains nodes that have more than one parents and that it can contain branches of different lengths. These problems motivate us to use graph transformer models [4] wherein we learn a transformer function on the label graph. As illustrated in Fig 4, the essential idea is to use this graph transformer to output a score map over those label nodes which then can be used to predict labels. This means that we also need to somehow incorporate input document representation into this graph neural network and that's where the need for a graph transformer model comes into the picture. The idea is to use that graph transformer to transform the label node representation and the input document representation into the score map over labels. Fig 5 illustrated how our graph transformer feature-propagation step works. Briefly, every step of the graph transformer consists of

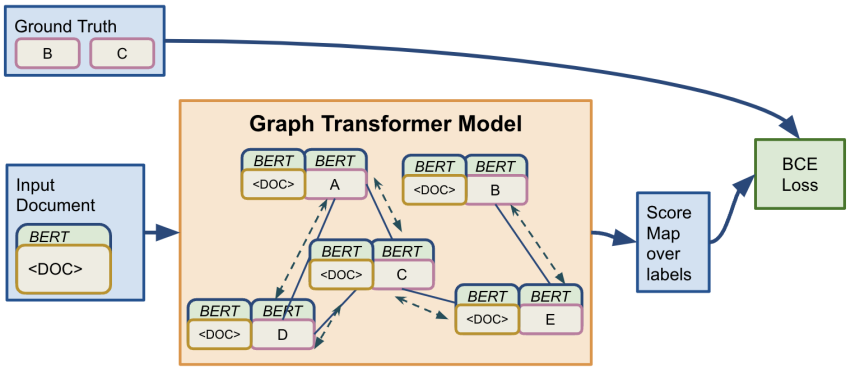

Figure 4: Graph Transformation architecture.

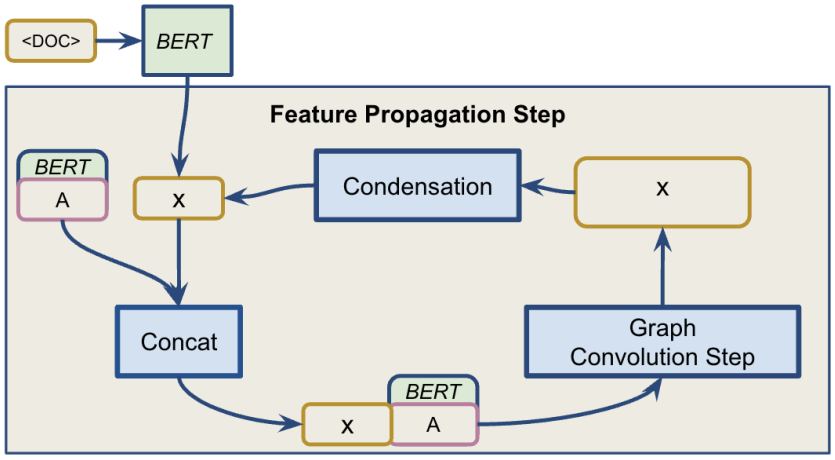

Figure 5: Feature propagation step in the graph transformer model.

three substeps-

$$Concat(X, L) : X = X||L$$
$$Convolution(X) : X = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}X$$
$$Condensation(X) : X = linear\_layer(X)$$

where X is initialized with the Bert encoding of the document text. L represents the Bert encoding of all the label nodes and A is the adjacency matrix corresponding to the label graph. We need a $Condensation$ substep because the concatenation step doubles the dimension size of X and the condensation step brings it back to its original size.

3 Experiments

3.1 Datasets

Our project utilized two key datasets. The first dataset is the Web of Science (WOS) dataset which contains a 2-level label hierarchy as shown in Figure 6. WOS is a collection of scientific documents paired with a 2-level label hierarchy structure represented as a tree. The first level represents the high-level classification of the document. For example, the document can be classified as a computer science document, medical science, etc. The second level classification is more granular and attempts to classify into a subarea of the level 1 classification. For example, if classified as CS, it will then attempt to classify whether it is a machine learning, computer graphics, or many other subfields of CS. Similarly for medical science, it will classify as cancer research, HIV research, or other subdomains

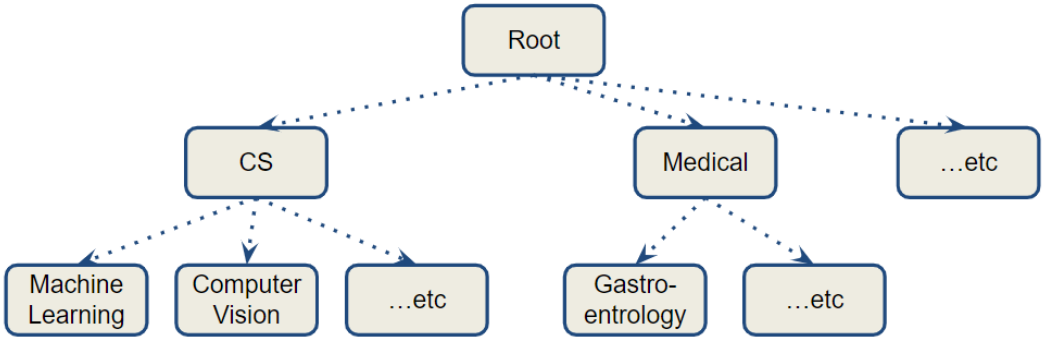

Figure 6: WOS Metadata Diagram

	4006	2407	3058	3763	6108	
concept_id	4006	2407	3058	3763	6108	
label	economic sector	regional planning	economic region	Romania	transition economy	r ec
alt_labels	[]	[inter-regional planning, regional management,...	[economic area, economic zone]	[]	[country in transition, economy in transition]	
parents	[]	[2402]	[]	[5087, 122, 914, 5781, 2200, 5283]	[5840]	

Figure 7: Eurlex child concept id's paired with a set of parents

of medical science. This dataset has about 31k documents and 134 finer and 7 coarser labels. We randomly split the dataset into train, val, and test split in the ratio 8:1:1.

The second dataset is the Eurlex dataset. In this dataset, there are approximately 57k legislative documents annotated with various labels from the EUROVOC concepts set. Of those 57k documents there are 47k in train split and 6k each in validation and test split. We utilized Eurlex because its label hierarchy cannot be represented as a tree, unlike WOS, making it a much more challenging dataset. For example, consider the label representation in Figure 7. The concept id refers to the current label and for some of the concept ids, there exist some parent labels. We utilized this hierarchy to reconstruct a DAG for our GNN in the form of a GNN matrix.

3.2 Evaluation Metrics

Since the WOS dataset is meant for a single-label classification setting and the Eurlex dataset for a multi-label classification setting, we also use different sets of evaluation metrics to compare these datasets.

WOS dataset. For WOS we use accuracy and MacroF1 as evaluation metrics. Accuracy is the ratio of the input examples on which the model predicts the correct label. MacroF1 is defined as the unweighted average of F1 scores calculated on per-label basis. F1 score is defined as

$$F1\_score(\tilde{y}, y) := \frac{2.\text{Precision}(\tilde{y}, y).\text{Recall}(\tilde{y}, y)}{\text{Precision}(\tilde{y}, y) + \text{Recall}(\tilde{y}, y)} \tag{1}$$

Eurlex dataset. For Eurlex we use Precision@1 and Recall@10 as evaluation metrics as has been done in [5]. Precision@1 calculates precision based on the top-1 predictions for every input while Recall@10 calculates recall based on top-10 predictions for every input.

3.3 Hyperparameters and other Training Details

In all of our models we use the `all-MiniLM-L6-v2` version of sentence transformer which is a BERT encoder that maps text to 384 dimension long vector. We use AdamW[6] optimizer with learning rate of 1e-3. We train all parameters of our model except that of the Bert's. In all of our models we use GELU[7] as our non-linear activation unit. We train all our models for 100 epochs but pick up that model which achieves the best score on the evaluation metrics as discussed in previous section

3.4 Results

The results on Table 1 demonstrate the performance of various classification models on the WOS dataset. Flat classification refers to simply classifying the documents directly into level 2 labels without utilizing the level 1 information. The key takeaway is that the proposed level-wise classification performed slightly worse than the flat classification. In order to compare both approaches fairly, we increased the number of parameters in the flat classifier so that they equal the number of parameters in the level-wise classifier. However, this resulted in the flat classifier having 3 hidden layers whereas the level-wise had just 2 layers. If we compared both models based on the number of layers, then we would expect level-wise to outperform flat classification because it has more tunable parameters. However, the GNN model described earlier outperformed other models. The best GNN model for the Eurlex dataset was the GNN with the bidirectional DAG where the number of convolutions was 2.

	Accuracy	Macro-F1
Flat Classification	0.585	0.545
Level-wise Classification	0.566	0.529
Graph Transformer (Bidirectional; 2 Convolutions)	0.611	0.571

Table 1: Accuracy and Macro-F1 of various classification models on the WOS dataset

The results on Table 2 demonstrate the performance of various classification models on the Eurlex dataset. Since the Eurlex label hierarchy is not based on levels, our levelwise approach is not applicable here. Instead, we simply compared our GNN approach with the flat approach. Given scalability constraints, we were unable to obtain complete results for GNN bidirectional which we believed would have outperformed Flat classification. The GNN model with a unidirectional DAG performed slightly worse than flat classification likely due to the limited scope of just a parent-to-child relationship of labels.

	Precision ($k = 1$)	Recall ($k = 10$)
Flat Classification	0.089	0.231
Graph Transformer (Unidirectional; 2 Convolutions)	0.056	0.231

Table 2: Precision @ $k = 1$ and Recall @ $k = 10$ of various classification models on the Eurlex dataset

The results on Table 3 demonstrate the performance of various classification models on the WOS data set with our GNN architecture using a unidirectional adjacency matrix. As stated above, the accuracy of classification on the WOS dataset significantly improved with the introduction of the novel GNN. The main hyperparameter that was tuned was the number of convolutions. When tuning

the number of convolutions, we observed no significant change in accuracy. This is because the path lengths of various nodes in the Eurlex DAG are generally short (a small number of nodes to get from one arbitrary node to another if there exists such a path). Thus, just a few convolutions are required to learn the relationships between a parent and a child along the path of the DAG.

Number of Convolutions	Accuracy	Macro F1
2	0.603	0.566
3	0.608	0.564
4	0.607	0.567
5	0.612	0.576

Table 3: Performance of Unidirectional Label DAG representation on WOS dataset (varying number of convolutions)

The results on Table 4 demonstrate the performance of various classification models on the WOS data set with our GNN architecture using a bidirectional adjacency matrix. Our results for the bidirectional representation are similar to the unidirectional representation results. When tuning the number of convolutions, a similar conclusion can be drawn wherein the GNN model saturates after a few convolutions given the shape of the bidirectional label hierarchy.

Number of Convolutions	Accuracy	Macro F1
2	0.611	0.571
3	0.600	0.554
4	0.601	0.558
5	0.601	0.559

Table 4: Performance of Bidirectional Label DAG representation on WOS dataset (varying number of convolutions)

4 Conclusions

In this paper, we propose a simple graph transformer model that incorporates both label-text and label-hierarchy to perform document classification. Our proposed model, although performs worse than flat classification on Eurlex dataset, far outperforms both the flat-classification model and the level-wise classification model on the WOS dataset.

5 Future Work

A critical drawback of our novel GNN model is that it does not perform well with the Eurlex dataset. We believe that with more tuning of the model, we can improve the performance of the GNN model on the Eurlex dataset and generalize our model to other XML datasets.

Additionally, label graphs can be very large and sparse for a large pool of labels. Since the DAG is the primary label representation in our GNN model, more work can be done to integrate label shortlisting algorithms with our GNN model to decrease the size of label graphs.

Acknowledgments

We thank Dr. Inderjit Dhillon who taught this course and Nilesh Gupta who was the TA for this course and provided invaluable guidance on our project.

References

[1] Kamran Kowsari et al. "HDLTex: Hierarchical Deep Learning for Text Classification". In: vol. abs/1709.08267. 2017. arXiv: 1709.08267. URL: http://arxiv.org/abs/1709.08267.

[2] Haibin Chen et al. "Hierarchy-aware Label Semantics Matching Network for Hierarchical Text Classification". 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)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 4370–4379. DOI: 10.18653/v1/2021.acl-long.337. URL: https://aclanthology.org/2021.acl-long.337.

[3] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: http://arxiv.org/abs/1908.10084.

[4] Vijay Prakash Dwivedi and Xavier Bresson. *A Generalization of Transformer Networks to Graphs*. 2021. arXiv: 2012.09699 [cs.LG].

[5] Yashoteja Prabhu and Manik Varma. "FastXML: A Fast, Accurate and Stable Tree-Classifier for Extreme Multi-Label Learning". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. New York, New York, USA: Association for Computing Machinery, 2014, pp. 263–272. ISBN: 9781450329569. DOI: 10.1145/2623330.2623651. URL: https://doi.org/10.1145/2623330.2623651.

[6] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. arXiv: 1711.05101 [cs.LG].

[7] Dan Hendrycks and Kevin Gimpel. *Gaussian Error Linear Units (GELUs)*. 2020. arXiv: 1606.08415 [cs.LG].