CheXGB: Combining Graph Neural Networks and BERT for automated radiology report labeling

Stanford CS224N Custom Project

Damir Vrabac
Department of Computer Science
Stanford University
dvrabac@stanford.edu

Viswesh Krishna
Department of Computer Science
Stanford University
viswesh@stanford.edu

Abstract

Healthcare systems wish to utilize the large quantities of unlabeled free-text radiology reports for training medical image models. Automated labelers allow healthcare systems to annotate tens of thousands of reports without expensive labor from doctors which would enable many hospitals around the world to train AI systems on their data. We propose CheXGB, an automated labeler that combines global information encoded by a heterogeneous graph of the free text reports and their associated words from a large chest X-ray data set (MIMIC-CXR) with local context information encoded by BERT. Using explicit global relations encoded by a graph neural network allows for inputs that purely NLP models are not trained to provide which is particularly useful in the data sparse regime we study. We find that variants of CheXGB outperforms CheXbert – the current state of the art in radiology report labeling – in 13 out of 14 classes and improve the average $kappa$ across tasks from 0.830 to 0.843.

1 Key Information to include

- Mentor: Akshay Smit
- External Mentor: Pranav Rajpurkar
- External Collaborators: Anirudh Joshi
- Sharing project: CS224W

2 Introduction

Text labeling is a well studied task in natural language processing that has widespread applicability [1, 2, 3, 4, 5, 6]. In medicine, text labeling has taken on increasing importance due to the lack of labeled data and the high cost of obtaining labels for data [7, 8]. Clinical text reports present in most fields of medicine provide a lot of information on the associated ground truth label for a given case which can be extracted by automated techniques without expensive human labor [9, 10]. CheXpert and CheXbert have studied heuristic and deep learning based approaches to the text labeling task in radiology for the goal of training Chest X-ray diagnosis models [10, 11]. The drawbacks of the two approaches noted above is their dependence on the creation of a heuristic based labeler for labeling previously unlabeled reports. Given that there are orders of magnitude more unlabeled reports than labeled reports, it is important for a labeler to be able to use the unlabeled reports effectively and to be able to attain good performance in a data sparse regime with few labeled data points.

CheXGB is motivated by the idea that graph neural networks can provide more information for the text labeling that traditional NLP models cannot provide. By using a graph that has edges between word-word and word-report, we ensure that in the second hop from a report node, we can access another report node. While implicit in NLP models, this global understanding of the dataset and the ability to connect related words that are not present in the same report but present in other similar

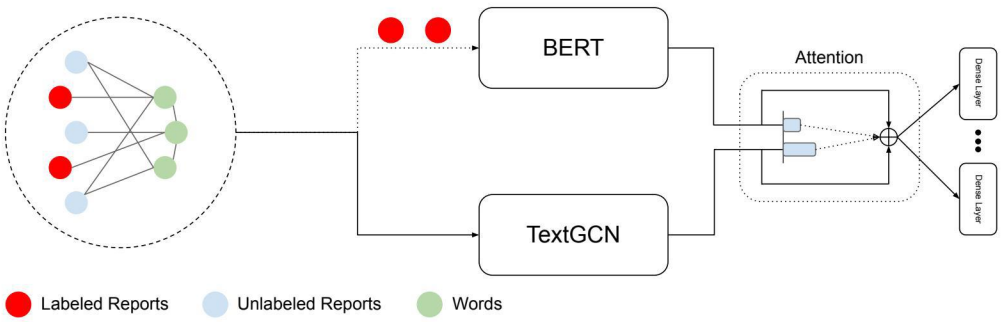

Figure 1: Overview of CheXGB. The input is a heterogeneous graph consisting of reports (both labeled and unalabeled) and words. First, all heterogeneous graph nodes are fed through TextGCN while only labeled reports are passed to BERT. Second, attention is performed on the output of BERT and the nodes corresponding to the labeled reports. Finally, the output of the attention layer is passed through a linear layer for multi-label class prediction.

reports is made explicit by Graph Neural Network models. In data sparse tasks, making the relations more explicit aids the learning process.

Our work provides the following contributions:

- A new model architecture and learning algorithm that leverages NLP and Graph Neural Network approaches to produce state of the art results on the radiology text labeling task.
- A comparison of how neighborhood sizes generated during subgraph sampling can influence the performance of Graph Neural Network based approaches.
- From an implementation standpoint, a custom data loader that can produce subgraphs from heterogeneous graphs and custom graph neural network layers that can consider edge weights during the aggregation process.

3 Related Work

3.1 NLP based Text Classification

Prior literature in NLP has focused on extracting structured labels from free text reports using heuristics and feature engineering [1, 2, 3, 4, 5, 6]. In the medical field, rules engines like NegEx and ontologies like ULMS (Unified Language Medical System) have been used to control vocabularies and identify dependencies relations between terms [7, 8]. In radiology, the CheXpert labeler is one such feature engineered labeler that relies on controlled extraction of medical mentions and a rule set for negation and uncertainty relations [10]. Recent work in deep learning has produced state of the art results within text classification. First recurrent and convolutional networks were used in extraction of labels from medical reports and more recently transformer architectures have produced the best results [6]. BERT and XLNet have been applied to extract normal and abnormal labels from thousands of radiology reports [5, 12]. Many of prior deep learning methods have focused on training solely on radiologist labeled data which is difficult and expensive to obtain [9]. CheXbert leverages benefits of both feature engineered labelers as well as end to end deep learning in a novel way to achieve state of the art results on label extraction from radiology reports [11]. CheXbert's primary limitation is that it relies on the feature engineered labeler to achieve state of the art results. While this is not an issue for datasets that do have feature engineered labelers, this would be a significant issue for any new dataset that is not structured similarly to CheXpert.

3.2 Graph Neural Networks for Text Classification

Prior literature in graph neural networks have focused on text classification tasks [13]. Yao et.al. propose a method called Text Graph Convolutional Network (TextGCN) which is a text classification method that leverages graph neural networks ability to preserve global structure information [14].

TextGCN applies Graph Convolutional Networks (GCN) proposed by Kipf and Welling [15] to a heterogeneous graph with words and documents as nodes. A major limitation of TextGCN is that the graph structure loses the sequential information inherent in text data. In other words, while TextGCN is able to represent global graph information it can only represent local sequential information in the form of co-occurrence weights between word-word edges.

3.3 BERT + Graph Neural Networks

A recent study had proposed augmenting BERT with a graph neural network [16]. The authors demonstrate that providing a vocabulary graph as an additional input to BERT can moderately improve BERT performance. However this study produced a graph that does not provide true global information and still emphasizes local information which is already encoded by BERT.

4 Methods

4.1 Graph Creation

As proposed by Yao et. al, we form a large heterogeneous graph consisting of both document and word nodes [14]. The total number of nodes in the graph is the sum of the number of both labeled and unlabeled documents $|D| \approx 200,000$ and the number of BERT embedding words $|B| \approx 30,000$. The node features for word nodes are set to BERT embeddings for the given word while the node features for document nodes are set to the output of pretrained BERT. We build two types of weighted edges in this heterogeneous graph: word-word edges based on point-wise mutual information (PMI) scores and document-word edges based on term frequency-inverse document frequency (TF-IDF) scores. Particularly, we calculate TF-IDF as

$$\text{TF-IDF}(d, w, D) = \text{TF}(d, w) \cdot \text{IDF}(w) = \left(\frac{\#d_w}{|d|)}\right)\left(\ln\left[\frac{1+|D|}{\frac{\#D_w}{|D|}+1}\right]\right) \tag{1}$$

where $\#d_w$ is the number of times the word w occurs in document d and $\#D_w$ is the number of documents in corpus D where w occurs. PMI is calculated with sliding windows over the corpus as follows:

$$\text{PMI}(w_1, w_2) = \log \frac{p(w_1, w_2)}{p(w_1)p(w_2)} \tag{2}$$

$$p(w_1, w_2) = \frac{\#W(w_1, w_2)}{\#W} \tag{3}$$

$$p(w) = \frac{\#W(w)}{\#W} \tag{4}$$

where $\#W$ is the number of sliding windows in the corpus, $\#W(x)$ is the number of sliding windows in the corpus where the word x exists in the sliding window and $\#W(x, y)$ is the number of sliding windows in the corpus where the words x and y co-occur. We use a sliding window of size 15 as for larger window sizes the test accuracy was not found to increase test accuracy as discussed in [14]. Further we only create sliding windows within single documents to contain contextual information from the PMI score within a single document. In order to use PMI as a weight in our graph we only include edges with positive PMI scores and thus filter out edges corresponding to small mutual information. Since positive PMI values imply a strong semantic correlation we only create edges between words with a positive PMI score. We also do not form self-loops unlike the graph creation approach used in [14]. Thus, the edge adjacency matrix A for the graph can be represented as:

$$A_{i,j} = \begin{cases} \text{PMI}(i, j) & i, j \text{ are words, PMI}(i, j) > 0 \\ \text{TF-IDF}(i, j) & i \text{ is document, } j \text{ is word} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

4.2 Architecture

4.2.1 GCN and GCNII as embedding layers to BERT

The Graph Convolutional Network (GCN) model was proposed in [15] and a single layer can be expressed as

$$H^{(l+1)} = \tilde{P}H^{(l)}W^{(l)} \tag{6}$$

where $\tilde{P}$ is a non-parametric function of the adjacency matrix, $H^{(l)}$ is the node features in the lth layer and $W^{(l)}$ is the node message matrix.

We use two GCN layers to aggregate neighborhood information of each subword with ReLU activations. The outputs of the GCN layers substitute the embedding layer in the BERT model and are then passed through the subsequent layers.

In addition, we implement a similar model substituting the GCN layers with GCNII layers proposed by Chen at. al [17]. GCNII generalizes the idea of residual connections to graph neural networks. The mathematical expression of the $l+1$-th layer for GCNII is

$$H^{(l+1)} = \left((1-\alpha)\hat{P}H^{(l)} + \alpha H^{(0)}\right)\left((1-\beta)I + \beta W^{(l)}\right) \tag{7}$$

where α is a hyperparameter controlling the residual connection to the initial node features and β is a hyperparameter controlling the node message function. Particularly, $\beta = 0$ corresponds to the message function being the identity.

4.2.2 ChexGB Architecture

We introduce ChexGB: a model architecture which incorporates learned node embeddings with the output of BERT through an attention layer and a prediction head illustrated in Figure 1. We use the TextGCN approach described in [14] to learn independent subword embeddings based on network information. This parallel architecture ensures that we maintain pretraining gains from BERT while incorporating neighborhood information in the final label prediction by simply adapting the prediction head. The concatenated output from TextGCN and BERT are fed into 14 linear layers that make classification on the 14 different radiology label tasks. Thus, each linear layer is modeled as a classification task of four classes: presence, absence, blank, or uncertain.

5 Experiments

5.1 Data

MIMIC-CXR v2.0.0 is a large dataset of chest x-rays and associated free-text reports with 180,000 imaging studies taken from 64,588 patients [18]. The labels are structured in a multi-label multi-class manner with each of the 14 classes having labels indicating presence, absence or uncertainty. Out of the approximately 180,000 report and x-ray pairs, 687 are labeled by board certified radiologists. We picked this dataset because it is one of the largest chest x-ray datasets in the world and has been well studied by many researchers in medical AI. This dataset also has several baselines benchmarked on it for text classification which makes it ideal for our project.

We choose a different dataset and split than that originally used in the CheXbert paper which contains sensitive patient information [11, 10]. We use 500 of the labeled reports in our training and hold out 150 reports for the test set.

5.2 Evaluation Metric

The overall task is text label extraction from radiology reports. The input is a radiology report and the output is 14 labels with one class for each label indicating presence, absence, uncertainty, or blank. We use a weighted κ metric across the 14 labels. For each label, we compute Cohen's κ – a measure of inter-rater similarity – which we weight based on the support for each label of interest.

Label	CheXbert	TextGCN				CheXGB			
		10	20	50	100	10	20	50	100
Enlarged Cardiomediastinum	0.682	0.044	0.232	0.349	0.211	0.581	0.712	0.551	**0.766**
Cardiomegaly	0.878	0.362	0.427	0.456	0.507	0.874	**0.911**	0.860	0.903
Lung Opacity	0.834	0.309	0.304	0.341	0.417	0.807	0.819	0.722	**0.844**
Lung Lesion	0.681	-0.033	0.234	-0.026	-0.026	0.693	0.718	**0.768**	0.648
Edema	0.930	0.522	0.547	0.426	0.390	0.919	**0.937**	0.895	0.937
Consolidation	0.827	0.207	0.474	0.359	0.376	0.918	**0.944**	0.837	0.843
Pneumonia	0.857	0.493	0.584	0.615	0.558	0.861	**0.897**	0.864	0.839
Atelectasis	0.741	0.237	0.384	0.353	0.371	0.765	0.734	**0.792**	0.778
Pneumothorax	0.935	0.588	0.762	0.710	0.558	0.961	0.952	**0.961**	0.961
Pleural Effusion	0.959	0.561	0.488	0.573	0.489	0.924	0.942	**0.945**	0.921
Pleural Other	0.814	0.00	-0.024	0.00	-0.019	0.628	**0.822**	0.814	0.140
Fracture	0.732	0.045	0.390	0.144	0.175	0.732	**0.732**	0.732	0.732
Support Devices	0.909	0.620	0.682	0.724	0.611	0.933	**0.928**	0.882	0.871
No Finding	**0.843**	0.635	0.618	0.691	0.732	0.711	0.756	0.781	0.756
Average	0.830	0.328	0.436	0.410	0.376	0.808	**0.843**	0.815	0.781

Table 1: Performance of CheXGB against state of the art baselines in NLP (CheXbert) for radiology report labeling and Graph Neural Networks (TextGCN) for text classification. We use neighborhood sampling sizes of 10, 20, 50, 100 for both TextGCN and CheXGB for evaluating model performance.

5.3 Comparisons

We choose the state of the art text labeling approaches from NLP (CheXbert) and Graph Neural Networks (TextGCN) to compare against as baselines. Since BERT is memory intensive and cannot run on all the reports in the graph simultaneously in one batch, we implement a custom version of neighbor sampling that can operate on heterogeneous graphs to create subgraphs based on hyperparameters such as the number of neighbors and the number of hops from a given hub node. Since we create batches based on the sub graph nodes, it is possible for a labeled report to be connected to an unlabeled report in two hops and thus we also supervise CheXGB only on the labeled reports. We generate four neighborhood sizes (10, 20, 50, and 100) and reference each version of CheXGB as CheXGB(n=number of neighbors sampled).

For our second methodological approach we use graph neural networks such as GCN and GCNII as embedding layers to the rest of BERT. We also compare against a custom version of GCNII with learnable residual connections as opposed to hyperparameters as originally described. In this approach the model learns how much weight to set to the residual connection during training.

5.4 Results

Comparison of the performance of the CheXGB model to the other state-of-the-art baselines can be found in Table 1.

We find that CheXGB with 20 neighbors ($n = 20$) outperforms CheXbert, the current state of the art radiology text labeler (0.843 vs 0.830 kappa on average task performance) and is the best performing model we studied. Studying the 14 task performances individually we find that variants of CheXGB with different sized neighbor samplings are the top performers across 13 out of the 14 tasks when compared to state of the art baselines in CheXbert and TextGCN. CheXGB with 20 neighbors performs the best on Cardiomegaly (0.911), Edema (0.937), Consolidation (0.944), Pneumonia (0.897), Pleural Other (0.822), Fracture (0.732), Support Devices (0.928). CheXGB with 50 neighbors performs the best on Lung Lesion (0.768), Atelectasis (0.792), Pneumothorax (0.961) and Pleural Effusion (0.945). CheXGB with 100 neighbors performs the best on Enlarged Cardiomegaly (0.766) and Lung Opacity (0.844). CheXbert is the top performer on only the No Findings (0.843) category.

5.4.1 Performance of GNN as an Embedding Layer

We find that using GNN models do not enable performance gains over the baseline. Using a GCN as an input embedding to CheXbert without the CheXGB parallel architecture harms performance and GCNII + CheXbert performs at par compared to the baseline CheXbert when a small signal of

the network is allowed ($\alpha = 0.9$ and $\beta = 0.0$). We also find that allowing α – the hyperparameter controlling residual connections to initial node features – to be a learned parameter also harms performance. This performance degradation is due to the introduction of the untrained GCN layers in the middle of the end-to-end pretrained CheXbert model. Particularly, since the CheXbert model trains the embedding layer and prediction head jointly, we hypothesize that the introduction of untrained GCN layers in between the embedding layer and prediction head significantly reduces the pretraining gains from BERT due to the change in the hidden activation distribution.

6 Analysis

6.1 Graph Statistics

Our graph consists of 230,000 nodes with 1.3 million edges between words and 13 million edges between reports and words. Looking at the connected components we find that apart from one large connected component that contains 194,000 nodes, the other connected components are disconnected nodes. These nodes are BERT vocabulary tokens that were never seen in any of the reports.

The average degree of all the reports is 69 and those of just the labeled reports is 113, indicating that CheXGB (n=100) contained nearly all of the nodes in the first hop neighbors.

Our top subwords in the graph sorted by degree correlate strongly to medically important terminology. We find subwords such as "card" (cardiomegaly, cardiomediastinum), "acute", "left", "pulmonary", "lung" and "pneumonia" all appear in the top 50 subwords by degree. This is important since it indicates that they are more likely to be sampled during subgraph generation since more reports and words are connected to them.

6.2 Neighborhood Size

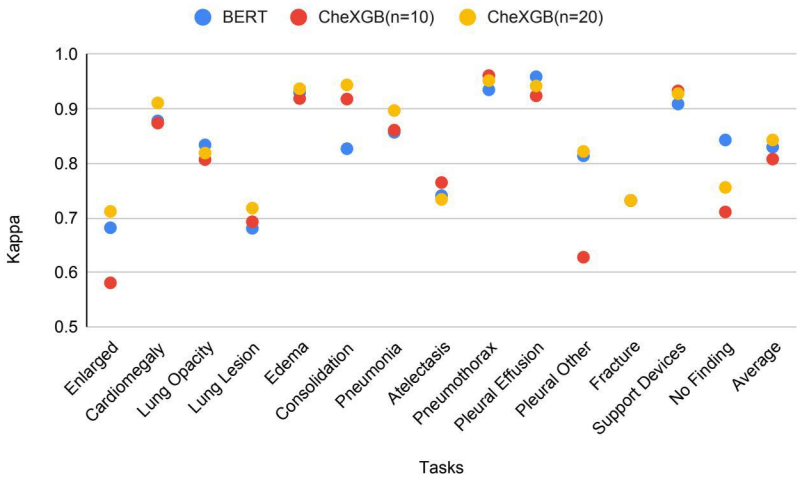

Figure 2: CheXGB with varying neighbor sizes compared to CheXbert performance. Increasing neighbor sampling size from 10 to 20 led to CheXGB outperforming CheXbert on 11 tasks.

From Figure 2 we see that scaling CheXGB from considering 10 neighbors during sampling to 20 neighbors during sampling helps it outperform CheXbert on several tasks. This improvement can be attributed to the model taking more advantage of the graph structure by considering more related

CheXbert	GCN + CheXbert	GCNII + CheXbert	Learned α + GCNII + CheXbert
0.830	0.154	0.816	0.654

Table 2: Performance of GCN and GCNII as embedding tables to CheXbert (score used is average κ)

words that are not directly present in the current impression but may be present in other impressions which are accessible due to graph neighborhood information. The increased neighbor size also allows CheXGB to consider more representations of unlabeled reports in the classification of labeled reports when contrasted with CheXbert which forms its representations while only considering the words present in the current impression.

6.3 Graph Parameter Efficiency

The majority of the parameters in CheXGB is contained within the BERT module. Specifically, the total amount of parameters in CheXGB is 125M of which 110M are attributed to BERT while 15M are attributed to TextGCN.

This also implies that during training, the bottleneck to scaling the subgraphs is BERT activations and not TextGCN. However with increased neighbor size, CheXGB activations scale proportionally while BERT does not. Future parameter efficient versions of BERT can be studied as part of CheXGB.

6.4 Training Speed

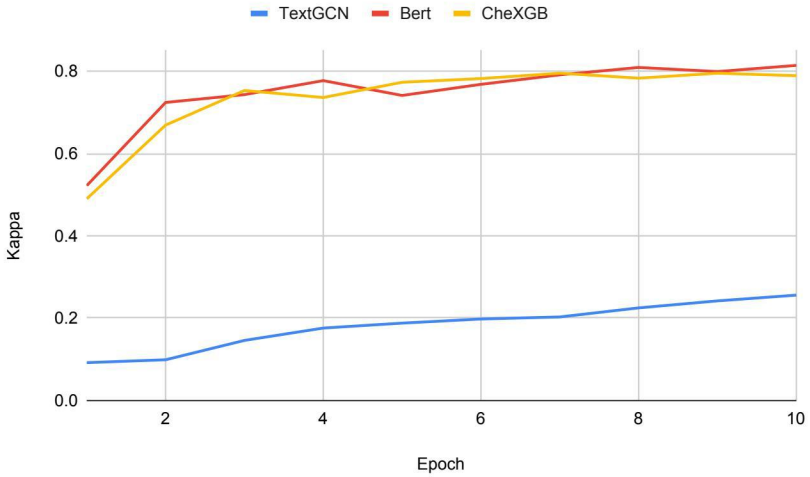

Figure 3: Performance of CheXGB, CheXbert and TextGCN attained by 10 epochs to understand training speed differences. CheXGB and CheXbert train at similar speeds while TextGCN takes longer to achieve its peak performance.

From Figure 3, we find that while TextGCN is slow to learn the task and requires 20+ epochs to reach peak performance, CheXbert and CheXGB are faster at learning and reach peak performance within the first 10 epochs.

6.5 Attention on Graph Representations

We find that CheXGB is attains near peak performance while leveraging both the graph representations and the BERT representations nearly equally. We see that variants of CheXGB use average attention weights of 0.42 on the graph representation and 0.58 on the BERT representation indicating that the graph contains information not directly present in BERT as seen in Figure 4.

7 Conclusion

We find that CheXGB achieves state of the art performance for radiology text labeling. The global information provided by the graph relations between reports and words that appear across reports, helps augment the local information that BERT is explicitly trained for. We find that increasing neighbor size can lead to improvements in neighbor performance up to 20 neighbors. We hypothesize

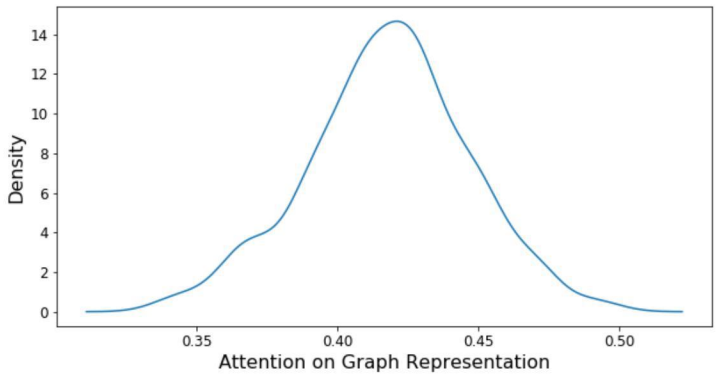

Figure 4: Distribution of the attention weights of variants of CheXGB model on the output of TextGCN from the test set.

that larger neighbor sizes need longer training periods to achieve better performance. Future work can investigate an ensemble of models as CheXbert still performs the best on the No Findings class and differing neighborhood sizes in CheXGB lead to some models performing the best on a subset of tasks.

8 Implementation

We implemented many custom components for our architecture and training loop by extending classes in DeepSnap and Torch Geometric. We implement a custom NeighborSampler for the purpose of generating heterogenous subgraphs from our large heterograph. This was necessary since BERT has high memory footprint and cannot run a batch size of 1000. We also implement a custom model architecture by extending DeepSnap to build a HeteroGCN that takes into account edge weights during message aggregation. We built the graph from scratch directly on the original dataset and made functions to codify edge weights through PMI and TFIDF.

References

[1] Marcin Michał Mirończuk and Jarosław Protasiewicz. A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, 106:36–54, 2018.

[2] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer, 2019.

[3] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[4] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE, 2017.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*, 2016.

[7] Saeed Mehrabi, Anand Krishnan, Sunghwan Sohn, Alexandra M Roch, Heidi Schmidt, Joe Kesterson, Chris Beesley, Paul Dexter, C Max Schmidt, Hongfang Liu, et al. Deepen: A negation detection system for clinical text incorporating dependency relation into negex. *Journal of biomedical informatics*, 54:213–219, 2015.

[8] Donald AB Lindberg, Betsy L Humphreys, and Alexa T McCray. The unified medical language system. *Methods of information in medicine*, 32(4):281, 1993.

[9] Matthew C Chen, Robyn L Ball, Lingyao Yang, Nathaniel Moradzadeh, Brian E Chapman, David B Larson, Curtis P Langlotz, Timothy J Amrhein, and Matthew P Lungren. Deep learning to classify radiology free-text reports. *Radiology*, 286(3):845–852, 2018.

[10] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 590–597, 2019.

[11] Akshay Smit, Saahil Jain, Pranav Rajpurkar, Anuj Pareek, Andrew Y Ng, and Matthew P Lungren. Chexbert: combining automatic labelers and expert annotations for accurate radiology report labeling using bert. *arXiv preprint arXiv:2004.09167*, 2020.

[12] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

[13] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637, 2018.

[14] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377, 2019.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[16] Zhibin Lu, Pan Du, and Jian-Yun Nie. Vgcn-bert: Augmenting bert with graph embedding for text classification, 2020.

[17] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks, 2020.

[18] Alistair E W Johnson, Tom J Pollard, Seth Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. Mimic-cxr: A large publicly available database of labeled chest radiographs. *arXiv preprint arXiv:1901.07042*, 2019.