An Empirical Study on Large-Scale Multi-Label Text Classification Including Few and Zero-Shot Labels

Ilias Chalkidis [†, ‡] **Manos Fergadiotis** [†, ‡] **Sotiris Kotitsas** [†]
Prodromos Malakasiotis [†, ‡] **Nikolaos Aletras** [*] **Ion Androutsopoulos** [†, ‡]
[†] Department of Informatics, Athens University of Economics and Business
[‡] Institute of Informatics & Telecommunications, NCSR "Demokritos"
[*] Computer Science Department, University of Sheffield, UK
[ihalk,fergadiotis,kotitsas,rulller,ion]@aueb.gr
n.aletras@sheffield.ac.uk

Abstract

Large-scale Multi-label Text Classification (LMTC) has a wide range of Natural Language Processing (NLP) applications and presents interesting challenges. First, not all labels are well represented in the training set, due to the very large label set and the skewed label distributions of LMTC datasets. Also, label hierarchies and differences in human labelling guidelines may affect graph-aware annotation proximity. Finally, the label hierarchies are periodically updated, requiring LMTC models capable of zero-shot generalization. Current state-of-the-art LMTC models employ Label-Wise Attention Networks (LWANs), which (1) typically treat LMTC as flat multi-label classification; (2) may use the label hierarchy to improve zero-shot learning, although this practice is vastly understudied; and (3) have not been combined with pre-trained Transformers (e.g. BERT), which have led to state-of-the-art results in several NLP benchmarks. Here, for the first time, we empirically evaluate a battery of LMTC methods from vanilla LWANs to hierarchical classification approaches and transfer learning, on frequent, few, and zero-shot learning on three datasets from different domains. We show that hierarchical methods based on Probabilistic Label Trees (PLTs) outperform LWANs. Furthermore, we show that Transformer-based approaches outperform the state-of-the-art in two of the datasets, and we propose a new state-of-the-art method which combines BERT with LWAN. Finally, we propose new models that leverage the label hierarchy to improve few and zero-shot learning, considering on each dataset a graph-aware annotation proximity measure that we introduce.

1 Introduction

Large-scale Multi-label Text Classification (LMTC) is the task of assigning a subset of labels from a large predefined set (typically thousands) to a given document. LMTC has a wide range of applications in Natural Language Processing (NLP), such as associating medical records with diagnostic and procedure labels (Mullenbach et al., 2018; Rios and Kavuluru, 2018), legislation with relevant legal concepts (Mencia and Fürnkranzand, 2007; Chalkidis et al., 2019b), and products with categories (Lewis et al., 2004; Partalas et al., 2015).

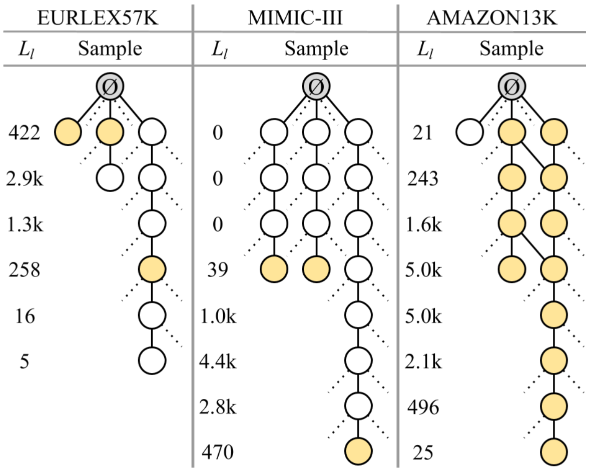

Figure 1: Examples from LMTC label hierarchies. $\emptyset$ is the root label. L_l is the number of labels per level. Yellow nodes denote gold label assignments. In EURLEX57K, documents have been tagged with both leaves and inner nodes (GAP: 0.45). In MIMIC-III, only leaf nodes can be used, causing the label assignments to be much sparser (GAP: 0.27). In AMAZON13K, documents are tagged with leaf nodes, but it is assumed that all the parent nodes are also assigned, leading to dense label assignments (GAP: 0.86).

Apart from the large label space, LMTC datasets often have skewed label distributions (e.g., some labels have few or no training examples) and a label hierarchy with different labelling guidelines (e.g., they may require documents to be tagged only with leaf nodes, or they may allow both leaf and other nodes to be used). The latter affects graph-aware annotation proximity (GAP), i.e., the proximity of the gold labels in the label hierarchy (see Section 4.1). Moreover, the label set and the hierarchies are periodically updated, thus requiring zero- and few-shot learning to cope with

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7503–7515,
November 16–20, 2020. ©2020 Association for Computational Linguistics

newly introduced labels. Figure 1 shows a sample of label hierarchies, with different label assignment guidelines, from three standard LMTC benchmark datasets: EUR-LEX (Chalkidis et al., 2019b), MIMIC-III (Johnson et al., 2017), and AMAZON (McAuley and Leskovec, 2013)).

Current state-of-the-art LMTC models are based on Label-Wise Attention Networks (LWANs) (Mullenbach et al., 2018), which use a different attention head for each label. LWANs (1) typically do not leverage structural information from the label hierarchy, treating LMTC as flat multi-label classification; (2) may use the label hierarchy to improve performance in few/zero-shot scenario, but this practice is vastly understudied; and (3) have not been combined with pre-trained Transformers.

We empirically evaluate, for the first time, a battery of LMTC methods, from vanilla LWANs to hierarchical classification approaches and transfer learning, in frequent, few, and zero-shot learning scenarios. We experiment with three standard LMTC datasets (EURLEX57K; MIMIC-III; AMAZON13K). Our contributions are the following:

- We show that hierarchical LMTC approaches based on Probabilistic Label Trees (PLTs) (Prabhu et al., 2018; Khandagale et al., 2019; You et al., 2019) outperform flat neural state-of-the-art methods, i.e., LWAN (Mullenbach et al., 2018) in two out of three datasets (EURLEX57K, AMAZON13K).

- We demonstrate that pre-trained Transformer-based approaches (e.g., BERT) further improve the results in two of the three datasets (EURLEX57K, AMAZON13K), and we propose a new method that combines BERT with LWAN achieving the best results overall.

- Finally, following the work of Rios and Kavuluru (2018) for few and zero-shot learning on MIMIC-III, we investigate the use of structural information from the label hierarchy in LWAN. We propose new LWAN-based models with improved performance in these settings, taking into account the labelling guidelines of each dataset and a graph-aware annotation proximity (GAP) measure that we introduce.

2 Related Work

2.1 Advances and limitations in LMTC

In LMTC, deep learning achieves state-of-the-art results with LWANs (You et al., 2018; Mullenbach et al., 2018; Chalkidis et al., 2019b), in most cases comparing to naive baselines (e.g., vanilla CNNs or vanilla LSTMs). The computational complexity of LWANs, however, makes it difficult to scale them up to extremely large label sets. Thus, Probabilistic Label Trees (PLTs) (Jasinska et al., 2016; Prabhu et al., 2018; Khandagale et al., 2019) are preferred in Extreme Multi-label Text Classification (XMTC), mainly because the linear classifiers they use at each node of the partition trees can be trained independently considering few labels at each node. This allows PLT-based methods to efficiently handle extremely large label sets (often millions), while also achieving top results in XMTC. Nonetheless, previous work has not thoroughly compared PLT-based methods to neural models in LMTC. In particular, only You et al. (2018) have compared PLT methods to neural models in LMTC, but without adequately tuning their parameters, nor considering few and zero-shot labels. More recently, You et al. (2019) introduced ATTENTION-XML, a new method primarily intended for XMTC, which combines PLTs with LWAN classifiers. Similarly to the rest of PLT-based methods, it has not been evaluated in LMTC.

2.2 The new paradigm of transfer learning

Transfer learning (Ruder et al., 2019; Rogers et al., 2020), which has recently achieved state-of-the-art results in several NLP tasks, has only been considered in legal LMTC by Chalkidis et al. (2019b), who experimented with BERT (Devlin et al., 2019) and ELMO (Peters et al., 2018). Other BERT variants, e.g. ROBERTA (Liu et al., 2019), or BERT-based models have not been explored in LMTC so far.

2.3 Few and zero-shot learning in LMTC

Finally, few and zero-shot learning in LMTC is mostly understudied. Rios and Kavuluru (2018) investigated the effect of encoding the hierarchy in these settings, with promising results. However, they did not consider other confounding factors, such as using deeper neural networks at the same time, or alternative encodings of the hierarchy. Chalkidis et al. (2019b) also considered few and zero-shot learning, but ignoring the label hierarchy.

Our work is the first attempt to systematically compare flat, PLT-based, and hierarchy-aware LMTC methods in frequent, few-, and zero-shot learning, and the first exploration of the effect of transfer learning in LMTC on multiple datasets.

3 Models

3.1 Notation for neural methods

We experiment with neural methods consisting of: (i) a *token encoder* ($\mathcal{E}_w$), which makes token embeddings (w_t) context-aware (h_t); (ii) a *document encoder* ($\mathcal{E}_d$), which turns a document into a single embedding; (iii) an optional *label encoder* ($\mathcal{E}_l$), which turns each label into a label embedding; (iv) a *document decoder* ($\mathcal{D}_d$), which maps the document to label probabilities. Unless otherwise stated, tokens are words, and $\mathcal{E}_w$ is a stacked BIGRU.

3.2 Flat neural methods

BIGRU-LWAN: In this model (Chalkidis et al., 2019b),[1] $\mathcal{E}_d$ uses one attention head per label to generate L document representations d_l:

$$a_{lt} = \frac{\exp(h_t^\top u_l)}{\sum_{t'} \exp(h_{t'}^\top u_l)}, \; d_l = \frac{1}{T}\sum_{t=1}^{T} a_{lt} h_t \quad (1)$$

T is the document length in tokens, h_t the context-aware representation of the t-th token, and u_l a trainable vector used to compute the attention scores of the l-th attention head; u_l can also be viewed as a label representation. Intuitively, each head focuses on possibly different tokens of the document to decide if the corresponding label should be assigned. In this model, $\mathcal{D}_d$ employs L linear layers with sigmoid activations, each operating on a different label-wise document representation d_l, to produce the probability of the corresponding label.

3.3 Hierarchical PLT-based methods

In PLT-based methods, each label is represented as the average of the feature vectors of the training documents that are annotated with this label. The root of the PLT corresponds to the full label set. The label set is partitioned into k subsets using k-means clustering, and each subset is represented by a child node of the root in the PLT. The labels of each new node are then recursively partitioned into k subsets, which become children of that node in the PLT. If the label set of a node has fewer than m labels, the node becomes a leaf and the recursion terminates. During inference, the PLT is traversed top down. At each non-leaf node, a multi-label classifier decides which children nodes (if any) should be visited by considering the feature vector of the document. When a leaf node is visited, the multi-label classifier of that node decides which labels of the node will be assigned to the document.

PARABEL, BONSAI: We experiment with PARABEL (Prabhu et al., 2018) and BONSAI (Khandagale et al., 2019), two state-of-the-art PLT-based methods. PARABEL employs binary PLTs ($k = 2$), while BONSAI uses non-binary PLTs ($k > 2$), which are shallower and wider. In both methods, a linear classifier is used at each node, and documents are represented by TF-IDF feature vectors.

ATTENTION-XML: Recently, You et al. (2019) proposed a hybrid method that aims to leverage the advantages of both PLTs and neural models. Similarly to BONSAI, ATTENTION-XML uses non-binary trees. However, the classifier at each node of the PLT is now an LWAN with a BILSTM token encoder ($\mathcal{E}_w$), instead of a linear classifier operating on TF-IDF document representations.

3.4 Transfer learning based LMTC

BIGRU-LWAN-ELMO: In this model, we use ELMO (Peters et al., 2018) to obtain context-sensitive token embeddings, which we concatenate with the pre-trained word embeddings to obtain the initial token embeddings (w_t) of BIGRU-LWAN. Otherwise, the model is the same as BIGRU-LWAN.

BERT, ROBERTA: Following Devlin et al. (2019), we feed each document to BERT and obtain the top-level representation h_{CLS} of BERT's special CLS token as the (single) document representation. $\mathcal{D}_d$ is now a linear layer with L outputs and sigmoid activations which operates directly on h_{CLS}, producing a probability for each label. The same arrangement applies to ROBERTA (Liu et al., 2019).[2]

BERT-LWAN: Given the large size of the label set in LMTC datasets, we propose a combination of BERT and LWAN. Instead of using h_{CLS} as the document representation and pass it through a linear layer with L outputs (as with BERT and ROBERTA), we pass all the top-level output representations of BERT into a label-wise attention mechanism. The entire model (BERT-LWAN) is jointly trained, also fine-tuning the underlying BERT encoder.

[1] The original model was proposed by Mullenbach et al. (2018), with a CNN token encoder ($\mathcal{E}_w$). Chalkidis et al. (2019b) show that BIGRU is a better encoder than CNNs. See also the supplementary material for a detailed comparison.

[2] Unlike BERT, ROBERTA uses dynamic masking, it eliminates the next sentence prediction pre-training task, and uses a larger vocabulary. Liu et al. (2019) reported better results in NLP benchmarks using ROBERTA.

3.5 Zero-shot LMTC

C-BIGRU-LWAN is a zero-shot capable extension of BIGRU-LWAN. It was proposed by Rios and Kavuluru (2018), but with a CNN encoder; instead, we use a BIGRU. In this method, $\mathcal{E}_l$ creates u_l as the *centroid* of the token embeddings of the corresponding label descriptor. The label representations u_l are then used by the attention heads.

$$v_t = \tanh(Wh_t + b) \tag{2}$$

$$a_{lt} = \frac{\exp(v_t^\top u_l)}{\sum_{t'} \exp(v_{t'}^\top u_l)}, \quad d_l = \frac{1}{T}\sum_{t=1}^{T} a_{lt} h_t \tag{3}$$

Here h_t are the context-aware embeddings of $\mathcal{E}_w$, a_{lt} is the attention score of the l-th attention head for the t-th document token, viewed as v_t (Eq. 2), d_l is the label-wise document representation for the l-th label. $\mathcal{D}_d$ also relies on the label representations u_l to produce each label probability p_l.

$$p_l = \text{sigmoid}(u_l^\top d_l) \tag{4}$$

The centroid label representations u_l of both encountered (during training) and unseen (zero-shot) labels remain unchanged, because the token embeddings in the centroids are not updated. This keeps the representations of unseen labels close to those of similar labels encountered during training. In turn, this helps the attention mechanism (Eq. 3) and the decoder (Eq. 4) cope with unseen labels that have similar descriptors with encountered labels.

GC-BIGRU-LWAN: This model, originally proposed by Rios and Kavuluru (2018), applies graph convolutions (GCNs) to the label hierarchy.[3] The intuition is that the GCNs will help the representations of rare labels benefit from the (better) representations of more frequent labels that are nearby in the label hierarchy. $\mathcal{E}_l$ now creates graph-aware label representations u_l^3 from the corresponding label descriptors (we omit the bias terms for brevity) as follows:

$$u_l^1 = f(W_s^1 u_l + \sum_{j \in N_{p,l}} \frac{W_p^1 u_j}{|N_{p,l}|} + \sum_{j \in N_{c,l}} \frac{W_c^1 u_j}{|N_{c,l}|}) \tag{5}$$

$$u_l^2 = f(W_s^2 u_l^1 + \sum_{j \in N_{p,l}} \frac{W_p^2 u_j^1}{|N_{p,l}|} + \sum_{j \in N_{c,l}} \frac{W_c^2 u_j^1}{|N_{c,l}|}) \tag{6}$$

$$u_l^3 = [u_l; u_l^2] \tag{7}$$

where u_l is again the centroid of the token embeddings of the descriptor of the l-th label; W_s^i, W_p^i, W_c^i are matrices for self, parent, and children nodes of each label; $N_{p,l}$, $N_{c,l}$ are the sets of parents and children of the the l-th label; and f is the tanh activation. The label-wise document representations d_l are again produced by $\mathcal{E}_d$, as in C-BIGRU-LWAN (Eq. 2–3), but they go through an additional dense layer with tanh activation (Eq. 8). The resulting document representations $d_{l,o}$ and the graph-aware label representations u_l^3 are then used by $\mathcal{D}_d$ to produce a probability p_l for each label (Eq. 9).

$$d_{l,o} = \tanh(W_o d_l + b_o) \tag{8}$$

$$p_l = \text{sigmoid}\left((u_l^3)^\top d_{lo}\right) \tag{9}$$

DC-BIGRU-LWAN: The stack of GCN layers in GC-BIGRU-LWAN (Eq. 5–6) can be turned into a plain two-layer Multi-Layer Perceptron (MLP), unaware of the label hierarchy, by setting $N_{p,l} = N_{c,l} = \emptyset$. We call DC-BIGRU-LWAN the resulting (deeper than C-BIGRU-LWAN) variant of GC-BIGRU-LWAN. We use it as an ablation method to evaluate the impact of the GCN layers on performance.

DN-BIGRU-LWAN: As an alternative approach to exploit the label hierarchy, we used a recent improvement of NODE2VEC (Grover and Leskovec, 2016) by Kotitsas et al. (2019) to obtain alternative hierarchy-aware label representations. NODE2VEC is similar to WORD2VEC (Mikolov et al., 2013), but pre-trains node embeddings instead of word embeddings, replacing WORD2VEC's text windows by random walks on a graph (here the label hierarchy).[4] In a variant of DC-BIGRU-LWAN, dubbed DN-BIGRU-LWAN, we simply replace the initial centroid u_l label representations of DC-BIGRU-LWAN in Eq. 5 and 7 by the label representations g_l generated by the NODE2VEC extension.

DNC-BIGRU-LWAN: In another version of DC-BIGRU-LWAN, called DNC-BIGRU-LWAN, we replace the initial centroid u_l label representations of DC-BIGRU-LWAN by the concatenation $[u_l; g_l]$.

GNC-BIGRU-LWAN: Similarly, we expand GC-BIGRU-LWAN with the hierarchy-aware label representations of the NODE2VEC extension. Again, we replace the centroid u_l label representations of GC-BIGRU-LWAN in Eq. 5 and 7 by the label representations g_l of the NODE2VEC extension. The resulting

[3] The original model uses a CNN token encoder ($\mathcal{E}_w$), whereas we use a BIGRU encoder, which is a better encoder. See the supplementary material for a detailed comparison.

[4] The NODE2VEC extension we used also considers the textual descriptors of the nodes, using an RNN encoder operating on token embeddings.

model, GNC-BIGRU-LWAN, uses both NODE2VEC and the GCN layers to encode the label hierarchy, thus obtaining knowledge from the label hierarchy both in a self-supervised and a supervised fashion.

4 Experimental Setup

4.1 Graph-aware Annotation Proximity

In this work, we introduce *graph-aware label proximity* (GAP), a measure of the topological proximity (on the label hierarchy) of the gold labels assigned to documents. GAP turns out to be a key factor in the performance of hierarchy-aware zero-shot capable extensions of BIGRU-LWAN. Let $G(L, E)$ be the graph of the label hierarchy, where L is the set of nodes (label set) and E the set of edges. Let $L_d \subseteq L$ be the set of gold labels a particular document d is annotated with. Finally, let $G_d^+(L_d^+, E_d^+)$ be the minimal (in terms of $|L_d^+|$) subgraph of $G(L, E)$, with $L_d \subseteq L_d^+ \subseteq L$ and $E_d^+ \subseteq E$, such that for any two nodes (gold labels) $l_1, l_2 \in L_d$, the shortest path between l_1, l_2 in the full graph $G(L, E)$ is also a path in $G_d^+(L_d^+, E_d^+)$. Intuitively, we extend L_d to L_d^+ by including additional labels that lie between any two assigned labels l_1, l_2 on the shortest path that connects l_1, l_2 in the full graph. We then define $\text{GAP}_d = \frac{|L_d|}{|L_d^+|}$. By averaging GAP_d over all the documents d of a dataset, we obtain a single GAP score per dataset (Fig. 1). When the assigned (gold) labels of the documents are frequently neighbours in the full graph (label hierarchy), we need to add fewer labels when expanding the L_d of each document to L_d^+; hence, GAP $\rightarrow 1$. When the assigned (gold) labels are frequently remote to each other, we need to add more labels ($|L_d^+| \gg |L_d|$) and GAP $\rightarrow 0$.

GAP should not be confused with *label density* (Tsoumakas and Katakis, 2009), defined as $\text{D} = \frac{1}{N} \sum_{d=1}^{N} \frac{|L_d|}{|L|}$, where N is the total number of documents. Although label density is often used in the multi-label classification literature, it is graph-unaware, i.e., it does not consider the positions (and distances) of the assigned labels in the graph.

4.2 Data

EURLEX57K (Chalkidis et al., 2019b) contains 57k English legislative documents from EUR-LEX.[5] Each document is annotated with one or more concepts (labels) from the 4,271 concepts of EUROVOC.[6] The average document length is approx. 727 words. The labels are divided in *frequent* (746 labels), *few-shot* (3,362), and *zero-shot* (163), depending on whether they were assigned to $n > 50$, $1 < n \leq 50$, or no training documents. They are organized in a 6-level hierarchy, which was not considered in the experiments of Chalkidis et al. (2019b). The documents are labeled with concepts from all levels (Fig. 1), but in practice if a label is assigned, none of its ancestor or descendent labels are assigned. The resulting GAP is 0.45.

MIMIC-III (Johnson et al., 2017) contains approx. 52k English discharge summaries from US hospitals. The average document length is approx. 1.6k words. Each summary has one or more codes (labels) from 8,771 leaves of the ICD-9 hierarchy, which has 8 levels (Fig. 1).[7] Labels are divided in *frequent* (4,112 labels), *few-shot* (4,216 labels), and *zero-shot* (443 labels), depending on whether they were assigned to $n > 5$, $1 < n \leq 5$, or no training documents. All discharge summaries are annotated with leaf nodes (5-digit codes) only, i.e., the most fine-grained categories (Fig. 1), causing the label assignments to be much sparser compared to EURLEX57K (GAP 0.27).

AMAZON13K (Lewis et al., 2004) contains approx. 1.5M English product descriptions from Amazon. Each product is represented by a title and a description, which are on average 250 words when concatenated. Products are classified into one or more categories (labels) from a set of approx. 14k. Labels are divided in *frequent* (3,108 labels), *few-shot* (10,581 labels), *zero-shot* (579 labels), depending on whether they were assigned to $n > 100$, $1 < n \leq 100$, or no training documents. The labels are organized in a hierarchy of 8 levels. If a product is annotated with a label, all of its ancestor labels are also assigned to the product (Fig. 1), leading to dense label assignments (GAP 0.86).

4.3 Evaluation Measures

The most common evaluation measures in LMTC are label precision and recall at the top K predicted labels ($P@K$, $R@K$) of each document, and $nDCG@K$ (Manning et al., 2009), both averaged over test documents. However, $P@K$ and $R@K$ unfairly penalize methods when the gold labels of a document are fewer or more than K, respectively. R-Precision@K ($RP@K$) (Chalkidis et al., 2019b), a top-K version of R-Precision (Manning et al.,

[5] http://eur-lex.europa.eu/

[6] http://eurovoc.europa.eu/

[7] www.who.int/classifications/icd/en/

	ALL LABELS		FREQUENT		FEW	
	RP@K	*nDCG@K*	*RP@K*	*nDCG@K*	*RP@K*	*nDCG@K*
EURLEX57K ($L_{AVG} = 5.07, K = 5$)						
FLAT NEURAL METHODS						
BIGRU-LWAN (Chalkidis et al., 2019b)	77.1	80.1	81.0	82.4	65.6	61.7
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	76.8	80.0	80.6	82.3	66.2	61.8
HIERARCHICAL PLT-BASED METHODS						
PARABEL (Prabhu et al., 2018)	78.1	80.6	82.4	83.3	59.9	57.3
BONSAI (Khandagale et al., 2019)	79.3	81.8	83.4	84.3	65.0	61.6
ATTENTION-XML (You et al., 2019)	78.1	80.0	81.9	83.1	68.9	64.9
TRANSFER LEARNING						
BIGRU-LWAN-ELMO (Chalkidis et al., 2019b)	78.1	81.1	82.1	83.5	66.8	61.9
BERT-BASE (Devlin et al., 2019)	79.6	82.3	83.4	84.6	69.3	64.4
ROBERTA-BASE (Liu et al., 2019)	79.3	81.9	83.4	84.4	67.5	62.4
BERT-BASE-LWAN (new)	**80.3**	**82.9**	**84.3**	**85.4**	**69.9**	**65.0**
MIMIC-III ($L_{AVG} = 15.45, K = 15$)						
FLAT NEURAL METHODS						
BIGRU-LWAN (Chalkidis et al., 2019b)	66.2	70.1	66.8	70.6	21.7	14.3
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	64.9	69.1	65.6	69.6	**35.9**	**21.1**
HIERARCHICAL PLT-BASED METHODS						
PARABEL (Prabhu et al., 2018)	58.7	63.3	59.3	63.7	9.6	6.0
BONSAI (Khandagale et al., 2019)	59.4	64.0	60.0	64.4	11.8	7.9
ATTENTION-XML (You et al., 2019)	**69.3**	**73.4**	**70.0**	**73.8**	26.9	19.5
TRANSFER LEARNING						
BIGRU-LWAN-ELMO (Chalkidis et al., 2019b)	66.8	70.9	67.5	71.3	21.2	13.0
BERT-BASE (Devlin et al., 2019)	52.7	58.1	53.2	58.4	18.2	10.0
ROBERTA-BASE (Liu et al., 2019)	53.7	58.9	54.3	59.2	18.1	10.9
BERT-BASE-LWAN (new)	50.1	55.2	50.6	55.5	15.3	9.1
AMAZON13K ($L_{AVG} = 5.04, K = 5$)						
FLAT NEURAL METHODS						
BIGRU-LWAN (Chalkidis et al., 2019b)	83.9	85.4	84.9	86.1	**80.0**	**73.6**
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	77.4	79.8	79.1	81.0	53.7	45.8
HIERARCHICAL PLT-BASED METHODS						
PARABEL (Prabhu et al., 2018)	85.1	86.7	86.3	87.4	76.8	71.9
BONSAI (Khandagale et al., 2019)	85.1	86.6	86.2	87.3	78.3	73.2
ATTENTION-XML (You et al., 2019)	84.9	86.7	86.0	87.4	76.0	69.7
TRANSFER LEARNING						
BIGRU-LWAN-ELMO (Chalkidis et al., 2019b)	85.1	86.6	86.2	87.4	79.9	73.5
BERT-BASE (Devlin et al., 2019)	86.8	88.5	88.5	89.6	70.3	62.2
ROBERTA-BASE (Liu et al., 2019)	84.1	85.9	85.7	87.0	70.6	61.3
BERT-BASE-LWAN (new)	**87.3**	**88.9**	**88.8**	**90.0**	77.2	68.9

Table 1: Results (%) of experiments across base methods for all, frequent, and few label groups. All base methods are incapable of zero-shot learning. The best overall results are shown in bold. The best results in each zone are shown underlined. We show results for K close to the average number of labels L_{AVG}.

2009), is better; it is the same as *P@K* if there are at least *K* gold labels, otherwise *K* is reduced to the number of gold labels. When the order of the top-*K* labels is unimportant (e.g., for small *K*), *RP@K* is more appropriate than *nDCG@K*.

4.4 Implementation Details

We implemented neural methods in TENSORFLOW 2, also relying on the HuggingFace Transformers library for BERT-based models.[8] We use the BASE versions of all models, and the Adam optimizer (Kingma and Ba, 2015). All hyper-parameters were tuned selecting values with the best loss on the development data.[9] For all PLT-based methods, we used the code provided by their authors.[10]

5 Results

5.1 Overall predictive performance

PLTS VS. LWANS: Interestingly, the TF-IDF-based PARABEL and BONSAI outperform the best previously published neural LWAN-based models on EURLEX57K and AMAZON13K, while being comparable to ATTENTION-XML, when all or frequent

[8] Consult https://tersorflow.org/ and http://github.com/huggingface/transformers/.

[9] See the appendix for details and hyper-parameters.

[10] PARABEL: http://manikvarma.org/code/Parabel/download.html; BONSAI: https://github.com/xmc-aalto/bonsai; ATTENTION-XML: http://github.com/yourh/AttentionXML

labels are considered (Table 1). This is not the case with MIMIC-III, where BIGRU-LWAN and ATTENTION-XML have far better results for all and frequent labels. The poor performance of the two TF-IDF-based PLT-based methods on MIMIC-III seems to be due to the fact that their TF-IDF features ignore word order and are not contextualized, which is particularly important in this dataset. To confirm this, we repeated the experiments of BIGRU-LWAN on MIMIC-III after shuffling the words of the documents, and performance dropped by approx. 7.7% across all measures, matching the performance of PLT-based methods.[11] The dominance of ATTENTION-XML in MIMIC-III further supports our intuition that word order is particularly important in this dataset, as the core difference of ATTENTION-XML with the rest of the PLT-based methods is the use of RNN-based classifiers that use word embeddings and are sensitive to word order, instead of linear classifiers with TF-IDF features, which do not capture word order. Meanwhile, in both EURLEX57K and AMAZON13K, the performance of ATTENTION-XML is competitive with both TF-IDF-based PLT-based methods and BIGRU-LWAN, suggesting that the bag-of-words assumption holds in these cases. Thus, we can fairly assume that word order and global context (long-term dependencies) do not play a drastic role when predicting labels (concepts) on these datasets.

Effects of transfer learning: Adding context-aware ELMO embeddings to BIGRU-LWAN (BIGRU-LWAN-ELMO) improves performance across all datasets by a small margin, when considering all or frequent labels. For EURLEX57K and AMAZON13K, larger performance gains are obtained by fine-tuning BERT-BASE and ROBERTA-BASE. Our proposed new method (BERT-BASE-LWAN) that employs LWAN on top of BERT-BASE has the best results among all methods on EURLEX57K and AMAZON13K, when all and frequent labels are considered. However, in both datasets, the results are comparable to BERT-BASE, indicating that the multi-head attention mechanism of BERT can effectively handle the large number of labels.

Poor performance of BERT on MIMIC-III: Quite surprisingly, all three BERT-based models perform poorly on MIMIC-III (Table 1), so we examined two possible reasons. First, we hypothesized that this poor performance is due to the distinctive writing style and terminology of biomedical documents, which are not well represented in the generic corpora these models are pre-trained on. To check this hypothesis, we employed CLINICAL-BERT (Alsentzer et al., 2019), a version of BERT-BASE that has been further fine-tuned on biomedical documents, including discharge summaries. Table 2 shows that CLINICAL-BERT performs slightly better than BERT-BASE on the biomedical dataset, partly confirming our hypothesis. The improvement, however, is small and CLINICAL-BERT still performs worse than ROBERTA-BASE, which is pre-trained on larger generic corpora with a larger vocabulary. Examining the token vocabularies (Gage, 1994) of the BERT-based models reveals that biomedical terms are frequently over-fragmented; e.g., 'pneumonothorax' becomes ['p', '##ne', '##um', '##ono', '##th', '##orax'], and 'schizophreniform becomes ['s', '##chi', '##zo', '##ph', '##ren', '##iform']. This is also the case with CLINICAL-BERT, where the original vocabulary of BERT-BASE was retained. We suspect that such long sequences of meaningless sub-words are difficult to re-assemble into meaningful units, even when using deep pre-trained Transformer-based models. Thus we also report the performance of SCI-BERT (Beltagy et al., 2019), which was pre-trained from scratch (including building the vocabulary) on scientific articles, mostly from the biomedical domain. Indeed SCI-BERT performs better, but still much worse than ATTENTION-XML.

Method	$\hat{T}$	$\hat{F}$	*nDCG*@15
ATTENTION-XML (You et al., 2019)	full-text	-	**73.4**
BERT-BASE (Devlin et al., 2019)	512	1.51	58.1
ROBERTA-BASE (Liu et al., 2019)	512	1.45	58.9
CLINICAL-BERT (Alsentzer et al., 2019)	512	1.60	58.6
SCI-BERT (Beltagy et al., 2019)	512	1.35	60.5
HIER-SCI-BERT (new)	4096	1.35	61.9

Table 2: Performance of BERT and its variants compared to ATTENTION-XML on MIMIC-III. $\hat{T}$ is the maximum number of (possibly sub-word) tokens used per document. $\hat{F}$ is the fragmentation ratio, i.e., the number of tokens (BPEs or wordpieces) per word.

A second possible reason for the poor performance of BERT-based models on MIMIC-III is that they can process texts only up to 512 tokens long, truncating longer documents. This is not a problem in EURLEX57K, because the first 512 tokens contain enough information to classify EURLEX57K documents (727 words on average), as shown by Chalkidis et al. (2019b). It is also not a problem in AMAZON13K, where texts are short (250 words on average). In MIMIC-III, however, the average document length is approx. 1.6k words and documents

[11] By contrast, the drop was less significant in the other datasets (4.5% in EURLEX57K and 2.8% in AMAZON13K).

	EURLEX57K ($K = 5$)		MIMIC-III ($K = 15$)		AMAZON13K ($K = 5$)	
	FEW ($n < 50$)	ZERO	FEW ($n < 5$)	ZERO	FEW ($n < 100$)	ZERO
BIGRU-LWAN (Chalkidis et al., 2019b)	61.7	-	14.3	-	**73.6**	-
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	51.0	33.5	15.0	31.5	9.9	20.8
DC-BIGRU-LWAN (new)	62.1	41.5	19.3	**39.3**	39.0	48.9
DN-BIGRU-LWAN (new)	52.2	23.8	10.0	22.3	20.4	27.2
DNC-BIGRU-LWAN (new)	62.0	39.3	**23.8**	33.6	41.6	47.6
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	61.8	**42.6**	21.1	35.2	45.8	46.1
GNC-BIGRU-LWAN (new)	**62.6**	36.3	18.4	34.2	45.3	**51.9**

Table 3: Results (%) of experiments performed with zero-shot capable extensions of BIGRU-LWAN. All scores are *nDCG@K*, with the same K values as in Table 1. Best results shown in bold. Best results in each zone shown underlined. n is the number of training documents assigned with a label. Similar conclusions can be drawn when evaluating with *RP@K* (See the appendix).

are severely truncated.[12] To check the effect of text truncation, we employed a hierarchical version of SCI-BERT, dubbed HIER-SCI-BERT, similar to the hierarchical BERT of Chalkidis et al. (2019a).[13] This model encodes consecutive segments of text (each up to 512 tokens) using a shared SCI-BERT encoder, then applies max-pooling over the segment encodings to produce a final document representation. HIER-SCI-BERT outperforms SCI-BERT, confirming that truncation is an important issue, but it still performs worse than ATTENTION-XML. We believe that a hierarchical BERT model pre-trained from scratch on biomedical corpora, especially discharge summaries, with a new BPE vocabulary, may perform even better in future experiments.

5.2 Zero-shot Learning

In Table 1 we intentionally omitted zero-shot labels, as the methods discussed so far, except GC-BIGRU-LWAN, are incapable of zero-shot learning. In general, any model that relies solely on trainable vectors to represent labels cannot cope with unseen labels, as it eventually learns to ignore unseen labels, i.e., it assigns them near-zero probabilities. In this section, we discuss the results of the zero-shot capable extensions of BIGRU-LWAN (Section 3.5).

In line with the experiments of Rios and Kavuluru (2018), Table 3 shows that GC-BIGRU-LWAN (with GCNs) performs better than C-BIGRU-LWAN in zero-shot labels on all three datasets. These two zero-shot capable extensions of BIGRU-LWAN also obtain better few-shot results on MIMIC-III comparing to BIGRU-LWAN; GC-BIGRU-LWAN is also comparable to BIGRU-LWAN in few-shot learning on EURLEX57K, but BIGRU-LWAN is much better than its two zero-shot extensions on AMAZON13K. The superior performance of BIGRU-LWAN on EURLEX57K and AMAZON13K, compared to MIMIC-III, is due to the fact that in the first two datasets few-shot labels are more frequent ($n \leq 50$, and $n \leq 100$, respectively) than in MIMIC-III ($n \leq 5$).

Are graph convolutions a key factor? It is unclear if the gains of GC-BIGRU-LWAN are due to the GCN encoder of the label hierarchy, or the increased depth of GC-BIGRU-LWAN compared to C-BIGRU-LWAN. Table 3 shows that DC-BIGRU-LWAN is competitive to GC-BIGRU-LWAN, indicating that the latter benefits mostly from its increased depth, and to a smaller extent from its awareness of the label hierarchy. This motivated us to search for alternative ways to exploit the label hierarchy.

Alternatives in exploiting label hierarchy: Table 3 shows that DN-BIGRU-LWAN, which replaces the centroids of token embeddings of the label descriptors of DC-BIGRU-LWAN with label embeddings produced by the NODE2VEC extension, is actually inferior to DC-BIGRU-LWAN. In turn, this suggests that although the NODE2VEC extension we employed aims to encode both topological information from the hierarchy and information from the label descriptors, the centroids of word embeddings still capture information from the label descriptors that the NODE2VEC extension misses. This also indicates that exploiting the information from the label descriptors is probably more important than the topological information of the label hierarchy for few and zero-shot learning generalization.

DNC-BIGRU-LWAN, which combines the centroids with the label embeddings of the NODE2VEC extension, is comparable to DC-BIGRU-LWAN, while being better overall in few-shot labels. Combining the GCN encoder and the NODE2VEC extension (GNC-BIGRU-LWAN) leads to a large im-

[12] In BPEs, the average document length is approx. 2.1k, as many biomedical terms are over-fragmented, thus only the 1/4 of the document actually fit in practice in BERT-based models.

[13] This model is 'hierarchical' in the sense that a first layer encodes paragraphs, then another layer combines the representations of paragraphs (Yang et al., 2016). It does not use the label hierarchy.

provement in zero-shot labels (46.1% to 51.9% *nDCG@K*) on AMAZON13K. On EURLEX57K, however, the original GC-BIGRU-LWAN still has the best zero-shot results; and on MIMIC-III, the best zero-shot results are obtained by the hierarchy-unaware DC-BIGRU-LWAN. These mixed findings seem related to the GAP of each dataset (Fig. 1).

The role of graph-aware annotation proximity: When gold label assignments are dense, neighbouring labels co-occur more frequently, thus models can leverage topological information and learn how to better cope with neighbouring labels, which is what both GCNs and NODE2VEC do. The denser the gold label assignments, the more we can rely on more distant neighbours, and the better it becomes to include graph embedding methods that conflate larger neighbourhoods, like NODE2VEC (included in GNC-BIGRU-LWAN) on AMAZON13K (GAP 0.86), when predicting unseen labels.

For medium proximity gold label assignments, as in EURLEX57K (GAP 0.45), it seems preferable to rely on closer neighbours only; hence, it is better to use only graph encoders that conflate smaller neighbourhoods, like the GCNs which apply convolution filters to neighbours up to two hops away, as in GC-BIGRU-LWAN (excl. NODE2VEC extension).

When label assignments are sparse, as in MIMIC-III (GAP 0.27), where only non-neighbouring leaf labels are assigned in the same document, leveraging the topological information (e.g., knowing that a rare label shares an ancestor with a frequent one) is not always helpful, which is why encoding the label hierarchy shows no advantage in zero-shot learning in MIMIC-III; however, it can still be useful when we at least have few training instances, as the few-shot results of MIMIC-III indicate.

Overall, we conclude that the GCN label hierarchy encoder does not always improve LWANs in zero-shot learning, compared to equally deep LWANs, and that depending on the proximity of label assignments (based on the label annotation guidelines) it may be preferable to use additional or no hierarchy-aware encodings for zero-shot learning.

6 Conclusions

We presented an extensive study of LMTC methods in three domains, to answer three understudied questions on (1) the competitiveness of PLT-based methods against neural models, (2) the use of the label hierarchy, (3) the benefits from transfer learning. A condensed summary of our findings is that (1) TF-IDF PLT-based methods are definitely worth considering, but are not always competitive, while ATTENTION-XML, a neural PLT-based method that captures word order, is robust across datasets; (2) transfer learning leads to state-of-the-art results in general, but BERT-based models can fail spectacularly when documents are long and technical terms get over-fragmented; (3) the best way to use the label hierarchy in neural methods depends on the proximity of the label assignments in each dataset. An even shorter summary is that no single method is best across all domains and label groups (all, few, zero) as the language, the size of documents, and the label assignment strongly vary with direct implications in the performance of each method.

In future work, we would like to further investigate few and zero-shot learning in LMTC, especially in BERT models that are currently unable to cope with zero-shot labels. It is also important to shed more light on the poor performance of BERT models in MIMIC-III and propose alternatives that can cope both with long documents (Kitaev et al., 2020; Beltagy et al., 2020) and domain-specific terminology, reducing word over-fragmentation. Pretraining BERT from scratch on discharge summaries with a new BPE vocabulary is a possible solution. Finally, we would like to combine PLTs with BERT, similarly to ATTENTION-XML, but the computational cost of fine-tuning multiple BERT encoders, one for each PLT node, would be massive, surpassing the training cost of very large Transformer-based models, like T5-3B (Raffel et al., 2019) and MEGATRON-LM (Shoeybi et al., 2019) with billions of parameters (30-100x the size of BERT-BASE).

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *EMNLP*. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv:2004.05150*.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019a. Neural legal judgment prediction in

English. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, Florence, Italy. Association for Computational Linguistics.

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019b. Large-Scale Multi-Label Text Classification on EU Legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, abs/1810.04805.

Philip Gage. 1994. A new algorithm for data compression. *C Users J.*, 12(2):23–38.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 855–864, New York, NY, USA. ACM.

Kalina Jasinska, Krzysztof Dembczynski, Róbert Busa-Fekete, Karlson Pfannschmidt, Timo Klerx, and Eyke Hüllermeier. 2016. Extreme F-measure Maximization Using Sparse Probability Estimates. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1435–1444. JMLR.org.

Alistair EW Johnson, David J. Stone, Leo A. Celi, and Tom J. Pollard. 2017. MIMIC-III, a freely accessible critical care database. *Nature*.

Sujay Khandagale, Han Xiao, and Rohit Babbar. 2019. Bonsai - Diverse and Shallow Trees for Extreme Multi-label Classification. *CoRR*, abs/1904.08249.

Diederik P. Kingma and Jim Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 5th International Conference on Learning Representations*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Sotiris Kotitsas, Dimitris Pappas, Ion Androutsopoulos, Ryan McDonald, and Marianna Apidianaki. 2019. Embedding Biomedical Ontologies by Jointly Encoding Network Structure and Textual Node Descriptors. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 298–308, Florence, Italy.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *J. Mach. Learn. Res.*, 5:361–397.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2009. *Introduction to Information Retrieval*. Cambridge University Press.

Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 165–172, New York, NY, USA. ACM.

Ryan McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep Relevance Ranking Using Enhanced Document-Query Interactions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Eneldo Loza Mencia and Johannes Fürnkranzand. 2007. An Evaluation of Efficient Multilabel Classification Algorithms for Large-Scale Problems in the Legal Domain. In *Proceedings of the LWA 2007*, pages 126–132, Halle, Germany.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *26th Int. Conf. on Neural Information Processing Systems*, Stateline, NV.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1101–1111. Association for Computational Linguistics.

Ioannis Partalas, Aris Kosmopoulos, Nicolas Baskiotis, Thierry Artières, Georgios Paliouras, Éric Gaussier, Ion Androutsopoulos, Massih-Reza Amini, and Patrick Gallinari. 2015. LSHTC: A Benchmark for Large-Scale Text Classification. *CoRR*, abs/1503.08581.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Conf. of NA Chapter of the Association for Computational Linguistics*, New Orleans, Louisiana, USA.

Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned Label Trees for Extreme Classification with Application to Dynamic Search Advertising. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 993–1002, Republic and Canton of Geneva, Switzerland.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-Shot and Zero-Shot Multi-Label Learning for Structured Label Spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What we know about how BERT works. *ArXiv*, abs/2002.12327.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15—-18.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv e-prints*.

Grigorios Tsoumakas and Ioannis Katakis. 2009. Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining*, 3:1–13.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. Association for Computational Linguistics.

Ronghui You, Suyang Dai, Zihan Zhang, Hiroshi Mamitsuka, and Shanfeng Zhu. 2018. AttentionXML: Extreme Multi-Label Text Classification with Multi-Label Attention Based Recurrent Neural Networks. *CoRR*, abs/1811.01727.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification. In *Advances in Neural Information Processing Systems 32*, pages 5812–5822. Curran Associates, Inc.

A Additional Implementation Details

All experiments were run in NVIDIA GT1080TI GPU cards, in a single GPU fashion. In Table 6, we report the size of the models and the elapsed training time. Hyper-parameters were tuned using HYPEROPT,[14] selecting values with the best loss on development data. Table 4 shows the hyper-parameters search space and the selected values. We use 200-D pretrained GLOVE embeddings (Pennington et al., 2014) for EURLEX57K and AMAZON13K, and 200-D WORD2VEC embeddings pre-trained on PUBMED[15] (McDonald et al., 2018) for MIMIC-III. For BERT-based methods we tuned only the learning rate, considering the values $\{2e\text{-}5, 3e\text{-}5, 5e\text{-}5\}$, selecting $2e$-5 for EURLEX57K and AMAZON13K, and $5e$-5 for MIMIC-III. Finally, for PARABEL and BONSAI we tuned the n-gram order in the range $\{1, 2, 3, 4, 5\}$, and the number of n-gram features in the range {100k, 200k, 300k, 400k}. When $n > 1$ we use n-grams up to order n, e.g. for $n = 3$ we use 1-grams, 2-grams and 3-grams. In all datasets the optimal values were 200k features for $n = 5$.

B BIGRUs vs. CNNs

Chalkidis et al. (2019b) showed that BIGRUs are better encoders than CNNs in EURLEX57K. We confirm these findings across all datasets (Table 5). BIGRU-LWAN, C-BIGRU-LWAN and GC-BIGRU-LWAN outperform CNN-LWAN, C-CNN-LWAN and GC-CNN-LWAN by 3.5 to 16.5 percentage points.

C Additional Results

Table 7 shows *RP@K* results of the zero-shot capable methods. As with *nDCG@K*, we conclude that the GCN label hierarchy encoder of Rios and Kavuluru (2018) does not always improve LWANs in zero-shot learning, compared to equally deep LWANs, and that depending on the proximity of label assignments, it may be preferable to use additional or no encodings of the hierarchy for zero-shot learning. Also, the zero-shot capable methods outperform BIGRU-LWAN in all, frequent, and few labels, but no method is consistently the best.

[14] https://github.com/hyperopt/hyperopt

[15] https://www.ncbi.nlm.nih.gov/pubmed/

			EURLEX57K		
Search space	Layers	Units	Dropout	Word Dropout	Batch Size
BASELINES	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
BIGRU-LWAN (Chalkidis et al., 2019b)	1	300	0.4	0	16
ZERO-SHOT	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	1	100	0.1	0.02	16
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	1	100	0.1	0	16
DC-BIGRU-LWAN (new)	1	100	0.1	0	16
DN-BIGRU-LWAN (new)	1	100	0.1	0	16
DNC-BIGRU-LWAN (new)	1	100	0.1	0	16
GNC-BIGRU-LWAN (new)	1	100	0.1	0.02	16
TRANSFER LEARNING	[12]	[768]	[0.1, 0.2, 0.3]	-	[8, 16]
BERT-BASE (Devlin et al., 2019)	12	768	0.1	-	8
ROBERTA-BASE (Liu et al., 2019)	12	768	0.1	-	8
BERT-LWAN (new)	12	768	0.1	-	8

			MIMIC-III		
Search space	Layers	Units	Dropout	Word Dropout	Batch Size
BASELINES	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
BIGRU-LWAN (Chalkidis et al., 2019b)	2	300	0.3	0	8
ZERO-SHOT	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	2	100	0.1	0	8
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	1	100	0.1	0	8
DC-BIGRU-LWAN (new)	1	100	0.1	0	8
DN-BIGRU-LWAN (new)	1	100	0.1	0	8
DNC-BIGRU-LWAN (new)	1	100	0.1	0	8
GNC-BIGRU-LWAN (new)	1	100	0.1	0	8
TRANSFER LEARNING	[12]	[768]	[0.1, 0.2, 0.3]	-	[8, 16]
BERT-BASE (Devlin et al., 2019)	12	768	0.1	-	8
ROBERTA-BASE (Liu et al., 2019)	12	768	0.1	-	8
BERT-LWAN (new)	12	768	0.1	-	8

			AMAZON		
Search space	Layers	Units	Dropout	Word Dropout	Batch Size
BASELINES	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
BIGRU-LWAN (Chalkidis et al., 2019b)	2	300	0.1	0	32
ZERO-SHOT	[1, 2]	[100, 200, 300, 400]	[0.1, 0.2, 0.3]	[0, 0.01, 0.02]	[8, 16]
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	2	100	0.1	0	32
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	1	100	0.1	0	32
DC-BIGRU-LWAN (new)	2	100	0.1	0	32
DN-BIGRU-LWAN (new)	1	100	0.1	0	32
DNC-BIGRU-LWAN (new)	2	100	0.1	0	32
GNC-BIGRU-LWAN (new)	1	100	0.1	0	32
TRANSFER LEARNING	[12]	[768]	[0.1, 0.2, 0.3]	-	[8, 16]
BERT-BASE (Devlin et al., 2019)	12	768	0.1	-	8
ROBERTA-BASE (Liu et al., 2019)	12	768	0.1	-	8
BERT-LWAN (ours)	12	768	0.1	-	8

Table 4: Hyper-parameter search space and best values chosen for all neural methods except BERT-based ones.

	All Labels		Frequent		Few		Zero	
	RP@K	$nDCG@K$	*RP@K*	$nDCG@K$	*RP@K*	$nDCG@K$	*RP@K*	$nDCG@K$
EURLEX57K ($L_{AVG} = 5.07, K = 5$)								
BIGRU-LWAN	**77.1**	**80.1**	**81.0**	**82.4**	**65.6**	**61.7**	-	-
CNN-LWAN	71.7	74.6	76.1	77.3	61.1	55.1	-	-
C-BIGRU-LWAN	**72.0**	**75.6**	**76.9**	**78.7**	**55.7**	**51.0**	**46.1**	**33.5**
C-CNN-LWAN	68.5	71.7	73.2	74.5	49.7	45.7	36.1	29.9
GC-BIGRU-LWAN	**76.8**	**80.0**	**80.6**	**82.3**	**66.2**	**61.8**	**48.9**	**42.6**
GC-CNN-LWAN	70.9	74.4	75.4	77.2	52.3	48.4	37.1	29.6
MIMIC-III ($L_{AVG} = 15.45, K = 15$)								
BIGRU-LWAN	**66.2**	**70.1**	**66.8**	**70.6**	**21.7**	**14.3**	-	-
CNN-LWAN	60.5	64.3	61.1	64.7	16.3	10.2	-	-
C-BIGRU-LWAN	**60.2**	**64.9**	**60.9**	**65.3**	**26.9**	**15.0**	**52.6**	**31.5**
C-CNN-LWAN	54.9	59.5	55.5	59.9	21.2	11.7	37.3	19.5
GC-BIGRU-LWAN	**64.9**	**69.1**	**65.6**	**69.6**	**35.9**	**21.1**	**56.6**	**35.2**
GC-CNN-LWAN	56.6	60.9	57.2	61.3	23.7	13.0	38.2	22.2
AMAZON13K ($L_{AVG} = 5.04, K = 5$)								
BIGRU-LWAN	**83.9**	**85.4**	**84.9**	**86.1**	**80.0**	**73.6**	-	-
CNN-LWAN	77.1	79.1	78.2	79.7	70.4	63.6	-	-
C-BIGRU-LWAN	**64.6**	**68.2**	**67.2**	**70.3**	**13.8**	**9.9**	**29.9**	**20.8**
C-CNN-LWAN	56.2	59.2	58.6	61.2	8.6	6.3	19.5	14.5
GC-BIGRU-LWAN	**77.4**	**79.8**	**79.1**	**81.0**	**53.7**	**45.8**	**56.1**	**46.1**
GC-CNN-LWAN	72.6	75.3	74.3	76.4	41.3	34.0	45.6	34.5

Table 5: Results (%) of experiments performed to compare GRU vs. CNN encoders. Best results in each zone shown in bold. We show results for K close to the average number of labels L_{AVG}.

Methods	Parameters	Trainable Parameter	Train Time
Baselines			
BIGRU-LWAN (Chalkidis et al., 2019b)	86	6	14h
Zero-shot			
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	80.2	0.2	9.3h
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	80.5	0.5	18.5h
DC-BIGRU-LWAN (new)	81.3	1.3	11.2h
DN-BIGRU-LWAN (new)	80.2	0.2	9.5h
DNC-BIGRU-LWAN (new)	81.6	1.6	10.1h
GNC-BIGRU-LWAN (new)	80.5	0.5	20.2h
Transfer learning			
BERT-BASE (Devlin et al., 2019)	110	110	9.5h
ROBERTA-BASE (Liu et al., 2019)	110	110	9.5h
BERT-LWAN (new)	119	119	11h

Table 6: Number of parameters (trainable or not) in millions and training time for a single run reported for all examined methods.

	EURLEX57K ($K = 5$)		MIMIC-III ($K = 15$)		AMAZON13K ($K = 5$)	
	Few ($n < 50$)	Zero	Few ($n < 5$)	Zero	Few ($n < 100$)	Zero
BIGRU-LWAN (Chalkidis et al., 2019b)	65.6	-	21.7	-	80.0	-
C-BIGRU-LWAN (Rios and Kavuluru, 2018)	55.7	46.1	26.9	52.6	13.8	29.9
DC-BIGRU-LWAN (new)	66.8	**53.9**	33.6	**63.9**	47.0	57.1
DN-BIGRU-LWAN (new)	56.9	34.3	19.5	43.9	27.1	36.9
DNC-BIGRU-LWAN (new)	66.9	51.7	**41.3**	59.4	50.2	59.6
GC-BIGRU-LWAN (Rios and Kavuluru, 2018)	66.2	48.9	35.9	56.6	53.7	56.1
GNC-BIGRU-LWAN (new)	**67.7**	49.4	31.6	57.5	**53.8**	**63.4**

Table 7: Results (%) of experiments performed with zero-shot capable extensions of BIGRU-LWAN. All scores are *RP@K*, with the same K values as in Table 1 of the main paper. Best results of zero-shot capable methods (excluding BIGRU-LWAN) shown in bold. Best results in each zone shown underlined. n is the number of training documents assigned with a label.