applied sciences

MDPI

Article

Integrating PubMed Label Hierarchy Knowledge into a Complex Hierarchical Deep Neural Network †

Stefano Silvestri *,‡, Francesco Gargiulo ‡ and Mario Ciampi

Institute for High Performance Computing and Networking, National Research Council of Italy (ICAR-CNR), Via Pietro Castellino 111, 80131 Naples, Italy; francesco.gargiulo@icar.cnr.it (F.G.); mario.ciampi@icar.cnr.it (M.C.)

* Correspondence: stefano.silvestri@icar.cnr.it

† This paper is an extended, improved version of the paper: F. Gargiulo, S. Silvestri and M. Ciampi, Exploit Hierarchical Label Knowledge for Deep Learning. In 2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS) 2019, Cordoba, Spain, 5–7 June 2019.

‡ These authors contributed equally to this work.

Abstract: This paper proposes an innovative method that exploits a complex deep learning network architecture, called Hierarchical Deep Neural Network (HDNN), specifically developed for the eXtreme Multilabel Text Classification (XMTC) task, when the label set is hierarchically organized, such as the case of the PubMed article labeling task. In detail, the topology of the proposed HDNN architecture follows the exact hierarchical structure of the label set to integrate this knowledge directly into the DNN. We assumed that if a label set hierarchy is available, as in the case of the PubMed Dataset, forcing this information into the network topology could enhance the classification performances and the interpretability of the results, especially related to the hierarchy. We performed an experimental assessment of the PubMed article classification task, demonstrating that the proposed HDNN provides performance improvement for a baseline based on a classic flat Convolution Neural Network (CNN) deep learning architecture, in particular in terms of hierarchical measures. These results provide useful hints for integrating previous and innate knowledge in a deep neural network. The drawback of the HDNN is the high computational time required to train the neural network, which can be addressed with a parallel implementation planned as a future work.

Keywords: extreme multilabel text classification; hierarchical deep neural network; natural language processing; BioBERT; PubMed MeSH

Citation: Silvestri, S.; Gargiulo, F.; Ciampi, M. Integrating PubMed Label Hierarchy Knowledge into a Complex Hierarchical Deep Neural Network. *Appl. Sci.* **2023**, *13*, 13117. https://doi.org/10.3390/app132413117

Academic Editor: Alessandro L. Koerich

Received: 31 October 2023
Revised: 27 November 2023
Accepted: 6 December 2023
Published: 9 December 2023

1. Introduction

The most recent studies in the field of artificial intelligence, and in particular in the Natural Language Processing (NLP) area, have proposed innovative deep learning (DL) network architectures, which were able to sensibly improve the previous state-of-the-art systems. Examples of these topologies are the Attention and Transformer-based Deep Neural Network architectures [1,2], which have provided a breakthrough in several application domains of medicine and healthcare, such as medical imaging or biomedical information extraction tasks [3–6].

On the other hand, among the various open issues related to DNNs, there is a need to incorporate prior and external knowledge directly into the neural architectures, allowing exploitation of predefined rules or knowledge by the neural architecture independently from its training [7]. Recently, some contributions to address this issue have been provided by Graph Neural Networks (GNNs) [8,9]. Moreover, it has been demonstrated that incorporating innate structures into artificial intelligence (AI) systems could provide substantial benefits [10].

Following these results, this paper presents a solution for incorporating an external innate structure into AI systems to integrate this kind of external knowledge within the DNN architecture. We applied the proposed approach to a specific difficult task: the

eXtreme Multilabel Text Classification (XMTC) [11,12]. This task consists of the labeling of textual documents with a variable number of labels belonging to a very large set, formed by thousands or more elements. In many real-world applications, the labels lie in a hierarchical tree-like structure, and in this case, the task can be called *hierarchical multilabel text classification*.

An example of a hierarchical XMTC task is the automatic assignment of the MeSH (Medical Subject Headings) terms to the biomedical scientific articles indexed by PubMed (https://www.ncbi.nlm.nih.gov/pubmed/ (accessed on 5 December 2023)), the official indexing and search engine for the biomedical scientific literature provided and maintained by the US National Library of Medicine (NLM). Domain experts manually tag each paper in PubMed for indexing and searching purposes. The classes used in PubMed belong to a large set called MeSH, where each label lies in a hierarchical tree structure. The manual annotation task is very long and difficult, and several methods have been proposed in the literature to support this process and implement a hierarchical extreme multilabel text classifier, like the ones described in the more recent BioASQ distributed tasks results [13–16]. The large-scale MeSH indexing task has previously been faced with hybrid DNNs and classic machine learning techniques [17], BERT-based solutions [18–20], or by adopting GNN-based approaches [21], improving the obtained performances over the years and often combining the integration of more approaches to exploit their respective advantages. Anyway, the complexity of this task and the need for more innovative and performing architectures is supported by the recent works presented in the literature, as well as by the distributed task proposed every year by BioASQ.

In this context, this paper proposes an innovative approach for the XMTC problem based on a Hierarchical Deep Neural Network (HDNN). This network topology internally reproduces the hierarchy of the labels within its structure. In this way, each node is specialized for the classification of its children nodes labels, considering both the results of the parent nodes and a representation of the input text. The network topology is dynamically and automatically obtained by an algorithm, which implements the network topology by iterating basic DNN blocks, following the hierarchical graph of the labels. In addition, to remove the hierarchical inconsistencies of the label set associated with each training sample, where some labels of the hierarchy could not be included [12], we also defined a methodology able to overcome this problem, expanding the label set following the hierarchy and adding the missing labels. We also evaluated the behavior and performances of the proposed HDNN approach applied to the PubMed MeSH indexing task, extending previous experiments presented in [22] and further demonstrating its effectiveness. In particular, this paper extends the previous results, first by including and testing the BioBERT model [18] as the initial layer of the proposed architecture for text encoding. BioBERT is a BERT-based architecture pretrained on a large biomedical domain document collection extracted from PubMed, which has been successfully adopted in many complex biomedical domain NLP tasks. Moreover, we more deeply analyzed the complexity of the algorithm, in particular with respect to the distribution of the number of nodes and the corresponding children nodes. Finally, we extended the number of experiments performed to further investigate the behavior of the proposed architecture.

The main novelty of the proposed approach is the development of a technique specifically devoted to incorporating hierarchical innate knowledge into the neural network architecture, able to dynamically adapt to any label set and capable of dealing with huge label numbers, such as the PubMed MeSH.

As mentioned above, the most recent best-performing approaches for large-scale PubMed indexing integrate more methodologies and techniques. Therefore, the proposed HDNN, combined with other approaches, can further improve the results obtained in this complex task. This paper extends and improves the research previously presented in [22], first integrating the BioBERT model [18] (a BERT model pretrained on a very large scientific biomedical article collection extracted from PubMed) within the HDNN architecture; moreover, it extends and more deeply investigates the preliminary experiments previously

presented. Finally, it provides a more accurate analysis of the performances of the proposed architecture, especially in terms of the depth of the label hierarchy.

This paper is structured as follows: The next Section 2 is devoted to a review of the most recent related works in the fields of integrating external knowledge into DNNs, hierarchical text classification, and PubMed large-scale indexing. In Section 3, the details of the methodology are described, and in Section 4, the experiments and results are illustrated and then discussed. Section 5 concludes the paper and presents some possible future works.

2. Related Works

Some interesting approaches to integrating prior knowledge have recently been presented in the literature. The work of [9] proposed a graph convolutional network that fuses external knowledge related to sentiment lexicon and part-of-speech information, with the purpose of improving the sentiment classification task. In detail, the authors exploited an external sentiment lexicon specifically developed for this purpose and used it to calculate a sentiment score for each word of the sentences of the dataset, building in this way a sentiment score matrix that weights the most important words for sentiment classification. The idea is to use this lexicon to compensate for the fact that the syntactic dependency trees cannot usually capture edge labels. Their experiments confirmed the effectiveness of the proposed model and also verified that the integration of external knowledge can support aspect-based sentiment analysis. In [8], the authors proposed a model to capture and exploit global dependencies among these sentences based on a graph neural network. Specifically, they first represented the input sentences as a graph, including various relations (entity–entity, sentence–sentence, and entity–sentence) to increase the information represented in the graph. After that, they introduced a graph recurrent network, which learns the semantic representations of the sentences. The experimental assessment showed significant performance improvements over the existing state-of-the-art models.

The research presented by [23] proposed to leverage preexisting class hierarchies, such as WordNet, to integrate additional domain knowledge into a DL classifier. They fitted the hierarchy of the classes into a probabilistic model and then derived a novel encoding of the labels and a corresponding loss function. Experiments on ImageNet and NABirds confirmed the effectiveness of this method.

In [24], a model named *KHGCN* is proposed, which is able to learn the entity relationship in the knowledge graph, removing the noisy information at the same time. This model is devoted to a personalized recommendation system. In detail, it extracts node embeddings in graphs and learns its hierarchical structure, also introducing an attentive mechanism to strengthen the knowledge graph aggregation. The experiments validated this on real-world datasets, demonstrating its effectiveness.

The authors of [25] presented a method for incorporating a label hierarchy into a DNN architecture, reproducing the same hierarchy in the neural network topology, to improve multiclass text classification tasks. In detail, the authors proposed an architecture named *HDLTex*, whose parent and child layers are connected following the exact topology of the hierarchical set of labels. The authors tested this method on a dataset composed of papers indexed on Web Of Science, comparing the results obtained by different combinations of neural network architectures (DNN, CNN, RNN) as nodes of the HDLTex model, demonstrating that this method can outperform baseline systems.

The idea of incorporating hierarchical class relations into a neural network has also been investigated by [26], where the authors applied a structured learning procedure incorporating hierarchical information using the labels, in addition to hierarchical distance, to successfully improve subpopulation shift modeling. The authors of [27] proposed a Topic-aware Hierarchical Multiple Attention Network for a hierarchical text classification model based on multihead self-attention integrated with convolutional filters to the end of capturing long-range dependencies. Moreover, this model also combines topic distributions generated by (LDA) Latent Dirichlet Allocation with sentence-level and document-level

inputs in a hierarchical architecture. The experimental results prove that this approach improves on the current state-of-the-art hierarchical models.

The application of DL to the medical domain can support the analysis of huge document collections but must face several problems, such as data acquisition, conservation and exchanging, imbalanced datasets, the need of explainability of the models, quality assurance, privacy preservation, respect of local laws, and other ethical issues [28].

The article presented by [29] describes a BiLSTM deep learning model based on a decision tree combined with a data balancing strategy, with the purpose of supporting automated diagnosis tasks from the analysis of questionnaires, also ensuring a secure data exchange between patients and their medical teams. The model also adopts a preliminary data preprocessing module, based on classic balancing algorithms to prepare the data for the network training to overcome imbalanced dataset issues. This issue has been also considered by [30], who proposed a DL method based on CNN combined with SMOTE to the end of effectively handling imbalanced data related to breast cancer. Their results demonstrate that this approach sensibly improves the binary classification of imbalanced datasets.

In [31], the XMTC task is tackled by a hybrid approach, which leverages two stages. The first one is devoted to the retrieval task and exploits two approaches, a Point Mutual Information method and a Unified Label-Semantic Embedding method, to the end of extracting hundreds of candidate labels from a huge label set. Then, the latter stage performs a ranking task, using a model based on a pretrained Transformer architecture, which is capable of distinguishing the true labels from the candidate ones and also providing an explainability mechanism. The experimental assessment showed performance improvements over the state-of-the-art methods.

The authors of [32] introduced a hierarchical Transformer-based architecture, specifically designed for the automatic diagnosis of eye diseases in textual ophthalmology electronic medical records. The proposed model includes a hierarchical Transformer architecture that learns the contextualized representations of each sentence in each document of the dataset, recognizing through an attention-based predictor the diagnosis parts of the documents and effectively extracting the required information related to the diseases.

The main difference between the proposed method here and the previously presented related works is the possibility of reproducing the hierarchical structure directly into the neural network architecture, dynamically adapting with any label hierarchy and also introducing prior knowledge into the neural network.

3. Methodology

The proposed HDNN for XMTC task is based on a neural network topology designed to reproduce the same hierarchical structure of the label set. To this end, we defined a recursive algorithm that automatically builds the network topology by iterating a basic DNN block, which corresponds to each node of the label hierarchy. In addition, we also exploited a methodology to expand the labels of each sample document of the training set, adding all missing ancestors in the label tree and obtaining a more consistent training dataset. In this section, the details of both these methods are described.

3.1. Hierarchical Deep Neural Network

The main purpose of the proposed methodology is the automatic creation of a DNN topology whose structure and interconnections reproduce the label hierarchy. To this end, a recursive algorithm is defined. This algorithm iterates basic neural network blocks, connecting them to each other following the label hierarchy, creating, as a result, the schema depicted in Figure 1. As shown in the figure, the nodes of the HDNN can be classified into three main groups:

- *Preprocessing node:* Where textual data in the input are converted into a word-embeddings-based representation (the green ones in the upper left part of the figure);
- *Internal node*: A DNN basic block whose inputs are word-embeddings-based and the classification result is obtained as the output of its parent node. In the same way, its output is connected in input to the corresponding children node;
- *Root Node*: A special internal node that corresponds to the root of the hierarchy. It differs from the other internal nodes because its input is directly and only connected to the output of the preprocessing node.

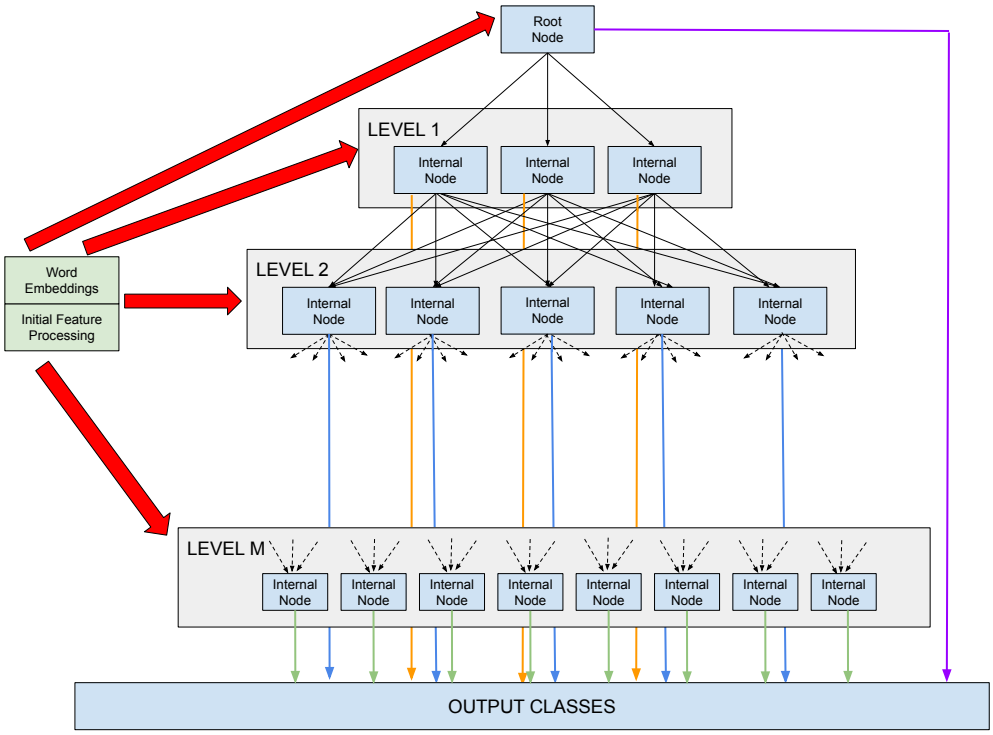

Figure 1. Schema of the internal architecture of an HDNN: The green blocks represent the mapping of the text into its own encoded version, provided to the root and all internal nodes (red arrows). The black arrows represent the outputs of each node, connected to the internal nodes of the next hierarchy level. The outputs of these nodes are also used to build the final classification output classes (depicted as colored arrows).

The preprocessing nodes create an encoded-word representation, exploiting static or dynamic word embedding models [2,33], exploiting in both cases models pretrained on biomedical document collections, to improve the final results [18,34]. The word vectors that represent the words of a document in input are extracted and then concatenated and convoluted in the block named *Initial Feature Processing*, producing a set of features extracted from the text representation, and sending it to each node of the HDNNs.

Figure 2a,b shows the details of the latter two aforementioned node types. These nodes have two main purposes: (i) specializing the features extracted for the corresponding level of labels in the hierarchy tree and (ii) predicting the classes and providing this information to their children nodes, in addition to the final complete prediction. The internal blocks of these nodes include a CNN layer, whose inputs are the results of the preprocessing block, followed by max-pooling and flattened layers; only in the case of internal nodes, the flattened output is concatenated with the output coming from its father node. Finally, the last layer is a dense fully connected layer devoted to label classification. As mentioned, the prediction is also sent to the children of that node in the topology. In summary, the nodes of one level specialize and extract the features for the same level of the hierarchy and provide as output the classifications, which are used as input by their corresponding children nodes.

The only structural difference between the root node and the internal nodes is that the latter concatenates the output of other directly connected internal nodes with the output of their own convolution layer. Furthermore, the convolution layer of the root node has a higher number of filters in order to capture more information at the top level of the topology.

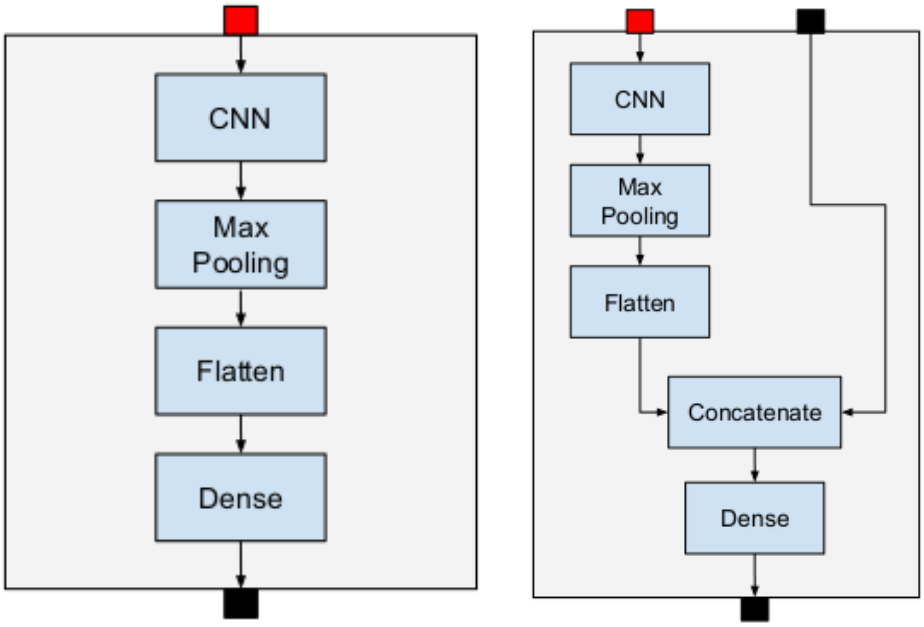

Figure 2. In (**a**), the **Root Node Structure** is depicted. The node inputs the encoded text (red input), which is processed using a CNN-dense internal structure. The result is the prediction of the first-level node children (black output). In (**b**), the **Internal Node Structure** is represented. The node takes in the encoded text (red input) and its parent node prediction (black input). The result is the prediction related to the subclassification problem corresponding to the node children (black output).

The topology of the HDNN can be fully automatically generated by iterating the aforementioned basic blocks, reproducing the same exact structure of the hierarchical graph of the labels. To this end, we specifically defined a recursive algorithm, whose details are shown in the next pseudocode in Algorithm 1. The pseudocode implements a recursive function whose input is the representation of the hierarchical label tree; the algorithm iterates the internal node layers blocks, connecting each node with both input text representation and all other children nodes, following the same structure of the hierarchy of the labels, obtaining in this way the architecture represented in Figure 1. The proposed algorithm can be used in any case where a predefined known hierarchical label structure is defined, allowing for the automatic network topology generation starting from the basic internal elements and the representation of the label hierarchy.

Algorithm 1 Pseudocode for HDDN generation

```
function MAKEMODEL(TREE, X1, X2)
    while tree : do
        children = tree.getChildren()
        if len(children)! = 0 then
            x2 = node(x1, x2, len(children))
            out = concatenate(out, x2)
            for child in children do
                makeModel(child, x1, x2)
```

3.2. Hierarchical Label Set Expansion

In real-world applications, it often happens that domain experts manually tag documents with labels belonging to a hierarchy without considering all the labels along a full path. In other words, a label set consistent with the hierarchy should be formed with all ancestors of each label in the corresponding tree, starting from the root. Manual tagging has mainly summary and indexing purposes and, thus, a more compact label set that does not consider all the labels along a whole path of the tree can better synthesize the document content. On the other hand, a machine learning (ML) system trained with a dataset with incomplete labeling in the perspective of the hierarchy can lead to unpredictable results. PubMed is an example of such a document collection: each manually assigned label belongs to a hierarchically organized set (MeSH), and the documents are not labeled with all the classes lying along the tree path formed by all the ancestors of the classes used as labels.

To improve the performances of machine learning systems trained on datasets like the ones described above, we proposed a methodology for the automatic expansion of the label set of each sample document of the training set, which adds any missing class along the hierarchy path for each label. We called this methodology *Hierarchical Label Set Expansion* (HLSE) [12]. Several experiments performed in previous studies confirmed the utility of the HLSE for the classification task, improving the final results. Moreover, considering the hierarchical network topology of the HDNN, it is necessary to include all the labels to reproduce a standard topology for each possible depth without missing some classes along the label set path. Therefore, it is necessary to apply the HLSE to all documents of the training set and test set before creating the corresponding HDNN when the data miss, in some cases, some labels along the hierarchy.

To better explain the HLSE, it is also important to describe more in detail the MeSH hierarchical tree. As mentioned above, the MeSH is a thesaurus provided and constantly updated by the NLM (National Library of Medicine), whose terms are hierarchically organized and controlled, as well as encoded with an alphanumeric code. The MeSH vocabulary is specifically designed to support the indexing of the biomedical scientific literature in PubMed, MEDLINE, and in the other NLM databases, as well as of any document belonging to the same area.

The MeSH terms are structured as 16 hierarchical trees, each one identified by a letter. These hierarchical trees have different levels, which include one or more subheadings, identified with their name and a numerical code, which starts with the same letter as their root node. Each deeper level in the same branch adds another numerical code to the higher-level code, separated by a period. Figure 3 shows an example of a part of this tree structure (the MeSH hierarchy is available and can be interactively browsed at https://meshb.nlm.nih.gov/treeView (accessed on 5 December 2023)) starting from the tree *Diseases* [C], with the corresponding class names and codes.

The colors of the blocks in Figure 3 can help us better clarify the issue that HLSE addresses, as well as better illustrate how it works. The right part of Figure 3 shows the full path from the root to the leaf *Edema, Cardiac*, also including both class names and MeSH codes (the code of the leaf is [C14.280.434.482]).

Usually, the MeSH tagging of the documents in PubMed does not include the full path, as shown in the Figure, but only some parts of it. For example, a human expert could tag a paper whose main topic is cardiac edema with the leaf *Edema, Cardiac* [C14.280.434.482] and another more generic label from a higher level, such as *Cardiovascular Diseases* [C.14] (in light blue in the figure). The expert could omit the other labels depicted in orange in the figure, namely *Diseases* [C], *Heart Diseases* [C.14.280], and *Heart Failure* [C.14.280.434], considering them unnecessary. While these missing labels might not be very useful for a human user, conversely, they could be very important in the training phase of a supervised machine learning model. If other papers on cardiac edema are tagged with the leaf *Edema, Cardiac* [C14.280.434.482] but with different labels along the hierarchy path, there will be high variability in the labels of documents with a very similar topic, making the correct training of an ML classifier very difficult.

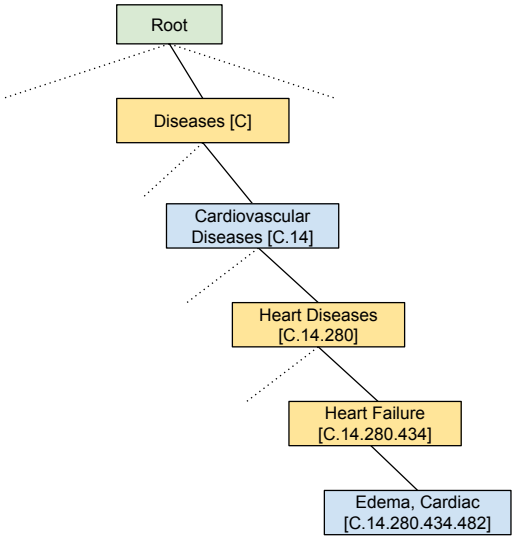

Figure 3. Example of a part of the MeSH label tree hierarchy that shows the full path from the root to the leaf *Edema, Cardiac*.

The HLSE has the purpose of removing this kind of noise from the training set, adding any missing label along the path from the root to each lower-level leaf in the labels of each document. Therefore, in the example in Figure 3, the labels in orange are added to the classes of that document. In this way, all the samples of the training set will have more uniform labeling. The drawback is an increase in the number of labels per document, but it was already demonstrated in [22] that this additional information improves hierarchical classification.

4. Experimental Assessment and Discussion

In this Section, we first describe the details and the features of the datasets used in the experimental assessment. Then, we illustrate the tools required to implement the proposed methods and the hardware used in the experiments. Then, an overview of the multilabel measures is provided, distinguishing flat and hierarchical measures. Finally, the obtained results are shown and discussed to try to investigate the behavior of the proposed HDNN in improving hierarchical classification when applied to the PubMed large-scale indexing XMTC problem.

4.1. Dataset

The experimental assessments have been performed on a subset of PubMed abstracts. We exploited the big data architecture described in [35] to download and extract from PubMed a collection of 11,075,577 abstracts, including the corresponding titles and their MeSH labels. Each document is tagged with a variable number of MeSHs, which currently counts more than 30,000 different labels. We also processed the labels of each document before applying HLSE (see Section 3.2) to complete the hierarchical label set of each document. It is important to underline that the same MeSH could be located in more branches of the tree, making the number of nodes of the tree higher than the total MeSH number, as shown in Table 1, obtaining 57,859 nodes after the use of the HLSE on the dataset. The automatic classification of a text collection with the above-described features is a hierarchical multiclass multilabel problem, as defined in Section 1. The dataset has been divided into a training set and a test set, respectively, randomly selecting 99% and 1% of the documents.

To simplify this complex problem, we decided to consider a maximum tree depth equal to five, substituting all the MeSHs that belong to a deeper level of the original hierarchy with the corresponding ancestor of the fifth level. In this way, we obtained 23,255 different tree nodes, distributed as shown in Figure 4, where it is possible to observe that a very high number of nodes have few children and very few nodes have more than 90 children. We

evaluated the average number of children per node to be 5.18, with a standard deviation of 7.30.

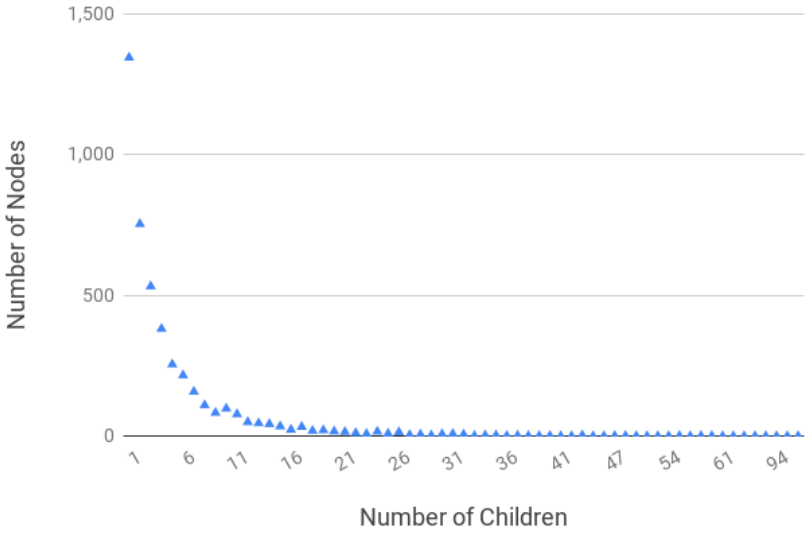

Figure 4. Distribution of the number of nodes per children node.

Table 1 summarizes the details of the datasets, showing the number of nodes, the average number of labels per document, and the average number of documents per label, and it compares these parameters considering a tree depth reduction to five and the original dataset with all levels, applying in both cases the HLSE.

Table 1. Dataset statistics: L is the number of tree nodes, L^* is the average labels count per document, and L^o is the average documents count per label.

MeSHDepth	L	L^*	L^o
5	23,255	39.18	18,740.05
All	57,859	50.57	19,614.76

The automatic classification of PubMed articles is included in the tasks of BioASQ (http://bioasq.org/ (accessed on 5 December 2023)), a research challenge on biomedical semantic indexing and question answering. A discussion of the experimental results can provide further details for improving the BioAsq results [13] and solving large XMTC problems.

4.2. Word Embedding Models

We tested two different embedding models to represent the input text. The first one is a classical static word embedding model [33] pretrained on a large biomedical-domain corpus formed by abstracts from PubMed articles, which is also provided as an additional resource for the BioASQ challenge (the model is publicly available at http://bioasq.org/news/bioasq-releases-continuous-space-word-vectors-obtained-applying-word2vec-pubmed-abstracts (accessed on 5 December 2023)). The latter model, namely the BioBERT model [18], leverages the dynamic word embedding representation extracted from a biomedical BERT model pretrained on a corpus formed by PubMed abstracts and PMC full-text articles, in addition to general-domain English Wikipedia and BooksCorpus data. The features of these models and the corresponding pretraining corpora details are reported in Tables 2 and 3.

Table 2. BioASQ word embedding features.

Vector Size	Training Algorithm	Window Size	Training Set Size	Vocabulary Size
200	CBOW	5	10,876,004	1,701,632

Table 3. BioBERT features.

BERT Model	Training Corpus Word Count	Biomedical Domain Corpus Word Count
BERT-large-cased	≈21.3 Billion words	≈18 Billion words

4.3. Tools and Hardware

Focusing on the tools used for the implementation and execution of the experimental assessment, the preprocessing phase (tokenization and sentence splitting) was performed using Apache Spark and Stanford CoreNLP wrappers for Spark (https://github.com/databricks/spark-corenlp (accessed on 5 December 2023)). The obtained NLP preprocessed text is the input of the word embedding layers, which converts the words into a vectorial representation. For this purpose, we used the Gensim Python library [36] version 4.2 in the case of static word embeddings, while we adopted the Hugging Face Python library (https://huggingface.co/ (accessed on 5 December 2023)) to obtain dynamic embeddings from the BioBERT model, providing a whole sentence as input.

The proposed HLSE and HDNN algorithms were also implemented in Python. In detail, we exploited Keras (https://keras.io (accessed on 5 December 2023)) version 2.11, with Tensorflow 2 backend for the definition, training, and testing of the HDNN. Moreover, the implementation requires the pyTree library, which provides a list-derived tree structure for Python, which is exploited to represent the label hierarchy. Finally, the Numpy, Pandas, and Pickle libraries were required to run the code. We underline that the code is publicly available on SoBigData research infrastructure (the code is publicly available at https://data.d4science.net/RgXa (accessed on 5 December 2023)).

All the experiments were performed on a deep learning cluster based on an IBM Power9 architecture, where each node was equipped with 2 Power9 CPUs at 3.7 GHz, 512 GB of RAM and 4 Nvidia Tesla V100 GPUs with 16 GB of dedicated VRAM. The operating system is Red Hat Enterprise Linux Server release 7.6. The current implementation of the HDNN runs on a single node and on a single GPU (the parallel implementation is not ready yet).

4.4. Evaluation Metrics

Different evaluation measures for multilabel text classification were proposed in the literature. These measures can be grouped into two classes: *flat* and *hierarchical* [37]. The flat measures base evaluates each single result obtained for each label in terms of Precision P_i, Recall R_i, and F1 score F_i for each class c_i, which equals

$$P_i = \frac{tp_{c_i}}{tp_{c_i} + fp_{c_i}}; \tag{1}$$

$$R_i = \frac{tp_{c_i}}{tp_{c_i} + fn_{c_i}} \tag{2}$$

$$F_i = \frac{2 \cdot P_i \cdot R_i}{P_i + R_i} \tag{3}$$

where tp_{c_i}, fp_{c_i}, and fn_{c_i} are, respectively, the true positives, false positives, and false negatives of the class c_i. These values are then averaged, obtaining the measures for the text classification task. The microaverage, which gives equal weight to each per-document classification result [38], is obtained by averaging over all labels each member of the fraction

of the previous equations, respectively, obtaining the micro-Precision MiP, micro-Recall MiR, and micro-F1 MiF, defined as

$$MiP = \frac{\sum_{i=1}^{L} tp_{c_i}}{\sum_{i=1}^{L}(tp_{c_i} + fp_{c_i})} \tag{4}$$

$$MiR = \frac{\sum_{i=1}^{L} tp_{c_i}}{\sum_{i=1}^{L}(tp_{c_i} + fn_{c_i})} \tag{5}$$

$$MiF = \frac{2 \cdot MiP \cdot MiR}{MiP + MiR} \tag{6}$$

with L equal to the total class number.

To complete the flat-measure-based evaluation, Accuracy Acc must be considered, which is calculated as

$$Acc = \frac{\sum_{i=i}^{M} \frac{t_{p_i} + t_{n_i}}{t_{p_i} + t_{f_i} + f_{p_i} + f_{n_i}}}{M} \tag{7}$$

where t_{n_i} are the true negatives for the class c_i.

Another flat measure is the example-based metrics, which evaluate bipartitions calculated on the average differences in the true label and the predicted label sets over all examples of the evaluation dataset. If we have a dataset with M multilabel examples and if Y_i, $i = 1 \ldots M$ is the set of true labels for each example and Z_i is the set of predicted labels for ith example, the example-based Precision, Recall, and F1 score, respectively, indicated as EBP, EBR, and EBF, are defined as

$$EBP = \frac{1}{M} \sum_{i=i}^{M} \frac{|Y_i \cap Z_i|}{|Z_i|} \tag{8}$$

$$EBR = \frac{1}{M} \sum_{i=i}^{M} \frac{|Y_i \cap Z_i|}{|Y_i|} \tag{9}$$

$$EBF = \frac{1}{M} \sum_{i=i}^{M} \frac{|Y_i \cap Z_i|}{|Y_i| + |Z_i|} \tag{10}$$

The EBP metric estimates how many predicted labels for the ith example are correct, while the EBR estimates the number of assigned labels correctly retrieved; EBF combines both measures for a global evaluation.

Furthermore, when the classification problem has a hierarchically organized label set, it is also possible to adopt the hierarchical measures that consider the label hierarchy in the classification results, taking into account the full path of the exact concepts in the class hierarchy and measuring whether parts of the path have been correctly classified. The definition of hierarchical measures is based on the augmented set of the true and predicted classes, respectively, Y_{aug} and $\hat{Y}_{aug}$, which are equal to

$$Y_{aug} = Y \cup An(y_1) \cup \cdots \cup An(y_N) \tag{11}$$

$$\hat{Y}_{aug} = \hat{Y} \cup An(\hat{y}_1) \cup \cdots \cup An(\hat{y}_M) \tag{12}$$

where $An(y_n)$ and $An(\hat{y}_m)$ are the ancestors of the true and predicted classes.

Starting from them, it is possible to define the hierarchical Precision, Recall, and F1 score (called HiP, HiR, and HiF). These measures consider the classifications result of a single document as a subtree path in the hierarchy as

$$HiP = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|\hat{Y}_{aug}|} \tag{13}$$

$$HiR = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|Y_{aug}|} \tag{14}$$

$$HiF = \frac{2 \cdot HiP \cdot HiR}{HiP + HiR} \tag{15}$$

The Y_{aug} and $\hat{Y}_{aug}$ identify two subtrees, respectively, representing the path of the true classifications and the path of the predicted classifications. The intersection of these two subtree paths is their similarity. Therefore, the HiP is the ratio between the subtree intersection and the predicted classifications paths, and the HiR is the ratio between the intersection and true classifications paths.

Adding all the ancestors in the sets, as in the previous case, overestimates the error if the hierarchy tree has nodes with many ancestors. To account for this behavior, the LCA-based measures [13] were defined. These kinds of measures are based on the *Lowest Common Ancestor* (LCA) concept from the graph theory [39]. The $LCA(n_1, n_2)$ of two nodes, n_1 and n_2, of a tree T is the node in T the furthest from the root that is an ancestor of both n_1 and n_2. To obtain LCA-based measures, the LCA concerning the set of predicted classes $\hat{Y}$ for each class y in the set of true classes Y, called $LCA(y, \hat{Y})$, and LCA for the set of true classes Y for each class $\hat{y}$ in the set of predicted classes $\hat{Y}$, named $LCA(\hat{y}, Y)$, must be calculated as

$$LCA(y, \hat{Y}) = \underset{m}{argmin} \cdot \gamma(m, y) \tag{16}$$

$$LCA(\hat{y}, Y) = \underset{m}{argmin} \cdot \gamma(m, \hat{y}) \tag{17}$$

where $\gamma(u, v)$ is the distance between the nodes u and v in the tree. Now, it is possible to define two graphs G_t and G_p, respectively, formed by the shortest paths from each $y \in Y$ to $LCA(y, \hat{Y})$ and the shortest path from each $\hat{y} \in \hat{Y}$ to $LCA(\hat{y}, Y)$. Then, the set Y_{aug_L} formed by all the nodes in the graph G_t and the set $\hat{Y_{aug_L}}$, which contains all the nodes in the graph G_p, must be constructed. Finally, LCA Precision $LCaP$, LCA Recall $LCaR$, and LCA F1 score $LCaF$ are defined as

$$LCaP = \frac{|\hat{Y}_{aug_L} \cap Y_{aug_L}|}{|\hat{Y}_{aug_L}|}; \tag{18}$$

$$LCaR = \frac{|\hat{Y}_{aug_L} \cap Y_{aug_L}|}{|Y_{aug_L}|} \tag{19}$$

$$LCaF = \frac{2 \cdot LCaP \cdot LCaR}{LCaP + LCaR} \tag{20}$$

In our experiments, whose main purpose is the contribution of the proposed HDNN to the incorporation of the label hierarchy knowledge directly into the neural network, the information provided by the hierarchical measures is very important.

4.5. Results and Discussion

In order to investigate the capability of the HDNN in incorporating the hierarchical label knowledge into the neural network, we compared it with a classic CNN architecture, such as the one described in [40], where the same CNN layers of the HDNN are implemented but organized in a flat topology. In detail, considering that we simplified the

problem using at least the fifth level of the MeSH tree in our experimental assessment, as described in Section 4.1, each internal node corresponding to an element of the hierarchy is formed by a cascade of two DNN layers (CNN and dense layers) for the two preprocessing CNN layers in the *Initial Feature Processing* block (see Figure 1). In this case, it is possible to estimate 17 layers for the deepest tree branch and 4 layers for the flat CNN architecture.

Tables 4 and 5, respectively, show the flat and hierarchical measures obtained with a simple CNN architecture and the proposed HDNN approach, using static (WE) or dynamic word embeddings (BioBERT). As expected, it is possible to note a performance improvement in the case of the HDNN, in both terms of flat and hierarchical measures. Moreover, the use of dynamic BioBERT-based embedding provides further improvement compared with the classic static word embedding approach in the first layer of the neural network. In more detail, the MiF and the EBF scores obtained by HDNN are, respectively, improved by 24.7% and 30.8% in the case of static WE and by 23.7% and 35.8% using BioBERT embeddings. Moreover, the increment in the HiF score is equal to 19.8% for HDNN WE and 35.9% for HDNN BioBERT. The higher impact of the HDNN is obtained in terms of LCA-based hierarchical measures: the LCa-F improves by 58.6% when static WEs are used and by 65.2% when exploiting the BioBERT model to encode input text.

In summary, the HDNN obtains improvements in overall performances in terms of all the considered measures (with the slight exception of HiP for HDNN WE, probably caused by the overestimation of the error provided by this measure in the case of nodes with many ancestors). More importantly, the results show a significant increment in the case of LCA-based hierarchical measures. The information related to this score considers the correct classification of the label hierarchy, giving a higher weight in the case of the assignment of a label that lies in the same hierarchy path. This behavior demonstrates that the HDNN is able to assign more correct labels with respect to a flat CNN architecture, but in many cases, other labels, although not originally included in the manual annotation, could be a reasonable assignment. The next examples can better clarify this result.

Referring to Figure 3, if a document related to cardiac edema is tagged with the label *Heart Failure* (the higher-level label in the hierarchy path), a flat measure will consider this result completely wrong, while the hierarchical measure will suggest that the result is not perfect but a good result. For our purposes, this information is very important because an improvement in the hierarchical measures shows that the neural network has improved the correct classification of the labels along the hierarchy path, demonstrating that some information about the hierarchy has been embedded within the HDNN.

More in detail, considering as an example a paper from the dataset, the article *Myocardial Edema: A Translational View* has been manually tagged with several MeSHs by NLM human experts (for simplicity, we focus on a single example). Among them, there is the label *Cell Enlargement* [G07.345.249.410.500]. The flat DNN architectures were not able to correctly predict it. On the other hand, the HDNN BioBERT, although it also did not find this label, identified a set of ancestors of this MeSH, namely *Physiological Phenomena* [G07], *Growth and Development* [G07.345], *Growth* [G07.345.249], and *Cell Growth Processes* [G07.345.249.410], which is the father node of the correct label. Analyzing the content of the paper, all these labels would not be considered wrong by a human annotator, as they are related to the topics of the scientific article. More importantly, the *Cell Growth Processes* [G07.345.249.410] MeSH is semantically very similar to its child *Cell Enlargement* [G07.345.249.410.500]. In summary, the HDNN approach suggested a set of possible correct candidate labels, and, moreover, these labels lie in the same tree of the correct MeSH, providing, as expected, an increment in hierarchical scores.

The obtained results demonstrate that the HDNN can support the integration of innate information and prior knowledge into a DNN and can be used to improve the performance in complicated tasks like XMTC.

Table 4. Performances of HDNN and CNN with static and dynamic embeddings in terms of flat measures.

Test	FLAT Measures						
	Acc	**EBP**	**EBR**	**EBF**	**MiP**	**MiR**	**MiF**
CNN WE	0.1353	0.7528	0.1398	0.2324	0.7522	0.1336	0.2268
CNN BioBERT	0.2344	0.7828	0.1619	0.2684	0.7810	0.1592	0.2644
HDNN WE	**0.1719**	**0.7578**	**0.1828**	**0.2894**	**0.7578**	**0.1739**	**0.2829**
HDNN BioBERT	**0.3015**	**0.8213**	**0.2187**	**0.3453**	**0.8191**	**0.2041**	**0.3270**

Table 5. Performances of HDNN and CNN with static and dynamic embeddings in terms of hierarchical measures.

Test	Hierarchical Measures					
	HiP	**HiR**	**HiF**	**LCa-P**	**LCa-R**	**LCa-F**
CNN WE	0.7839	0.1635	0.2665	0.5530	0.0738	0.1273
CNN BioBERT	0.8117	0.1775	0.2899	0.5824	0.1113	0.1819
HDNN WE	0.7820	**0.2052**	**0.3194**	**0.6350**	**0.1239**	**0.2020**
HDNN BioBERT	**0.8229**	**0.2573**	**0.3941**	**0.7177**	**0.1944**	**0.3005**

On the other hand, a limitation of the proposed approach is related to its high computational time. The training time required in our experiments is approximately three days for each model, while the prediction on the test set and the calculation of the evaluation measure need approximately 15 min. The flat CNN model requires approximately 1 h to be trained on the same dataset and a few minutes for the prediction task. More in detail, we measured the training times of both flat CNN and HdNN approaches, observing in the case of HDNN 15,000 s for each epoch formed by 100,000 document samples, whereas the CNN training time for the same epoch is about 100 s, using the hardware described in Section 4.1. The HDNN training time is very high, especially if compared with the other flat CNN architecture, although both network topologies are characterized by a comparable number of parameters: 40,079,550 in the case of CNN and 45,194,480 in the case of HDNN. The very high training time of the HDDN is caused by the complexity of the graph topology, where a large number of interconnections among all the nodes are involved in reproducing the label tree structure. Moreover, each internal node of the HDNN has to wait for the outputs of its ancestor nodes before proceeding to execute forward and backward passes during the Stochastic Gradient Descent of the training of the model.

We compared the training time of a CNN and an HDNN with an increasing depth of labels in the training set for a deeper analysis of this behavior. We observed a comparable training time for CNN and HDNN in the case of labels belonging only to the first layer of the hierarchy. On the other hand, the training time of the HDNN increases in an exponential way at each deeper level of the hierarchy we included in the training set, while the CNN does not show substantial increases in training time.

5. Conclusions

This paper presented an innovative deep learning network architecture based on a hierarchical structure, named *Hierarchical Deep Neural Network* (HDNN), specifically devoted to the hierarchical XMTC problem. The topology of the HDNN reproduces the same hierarchical structure of the labels. This general concept can be applied in all multilabel classification problems where the labels are organized in a hierarchical structure.

After an introduction to the problem and a related works section, we described the details of the HDNN and a methodology for the automatic creation of this DNN,

starting from the label hierarchy structure, based on a recursive algorithm that iterates basic DNN blocks. We also defined a method for the regularization of the training set, named Hierarchical Label Set Expansion (HLSE), which adds to each document label set the corresponding ancestors if they are not present, obtaining a training set consistent with the predefined hierarchy.

The experimental assessment focused on the automatic labeling of the biomedical scientific literature indexed in PubMed, which adopts the MeSHs (Medical Subjects Headings) hierarchical label set. The obtained results can be useful for the improvement of the large biomedical indexing task, integrating it in more complex methodologies, as proposed by the most recent methodologies in the BioASQ distributed challenges [13–16], which usually integrate more methodologies to obtain state-of-the-art performances.

A limit of the proposed methodology is its high computation time, mainly required to update the network weights, caused by the need to wait for the complete update of the upper layers' weights before the calculation of the corresponding children nodes' new neural network weights. This issue can be addressed by adopting more powerful GPU architecture, such as the newer Nvidia H100 architecture, as well as implementing a parallel version of the HDNN training routine, capable of distributing the computational complexity among different nodes and GPUs. We envisage the development of a parallel version of the HDNN code as a future work. A possible strategy for the parallel implementation includes the development of a multi-GPU code, which runs each internal node of the same level of the HDNN on separate GPUs in parallel. Another possibility is to adapt some GNN parallel approaches, such as the ones proposed in [41–43], allowing better exploitation of the features of modern multi-GPU hardware architectures.

On the other hand, the results obtained in our experiments show performance improvements, especially in the cases of hierarchical measures. These results confirm that the HDNN is capable of improving the correct classification of the hierarchy of the assigned labels.

In future work, in addition to the parallel implementation of the code, we are planning to perform more complex experiments with more depth classification cases. We also want to formalize the network complexity correlated to the number of connections and parameters and the network depth to better define the problem and analyze its complexity. We are also studying methods to reduce the complexity of the network by changing the interconnections of the layers and adopting alternative training strategies. We are also planning new experiments considering different kinds of internal nodes of the HDNN, such as attention-based networks and other DNN architectures. Finally, given the complexity involved, we are also considering evaluating how this methodology can fit the newest context of quantum computing [44].

Author Contributions: Conceptualization, S.S. and F.G.; methodology, S.S. and F.G.; software, S.S. and F.G.; validation, S.S., F.G. and M.C.; formal analysis, S.S. and F.G.; investigation, S.S. and F.G.; resources, S.S., F.G. and M.C.; data curation, S.S. and F.G.; writing—original draft preparation, S.S. and F.G.; writing—review and editing, S.S., F.G. and M.C.; visualization, S.S. and F.G.; supervision, M.C.; project administration, M.C.; funding acquisition, M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by European Union—NextGenerationEU—National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR)—Project: "SoBigData.it—Strengthening the Italian RI for Social Mining and Big Data Analytics"—Prot. IR0000013—Avviso n. 3264 del 28/12/2021.

Data Availability Statement: The HDNN model code and the dataset are shared on the SoBigData research infrastructure https://data.d4science.net/RgXa (accessed on 5 December 2023).

Acknowledgments: We thank Giuseppe Trerotola and Simona Sada for the administrative support they provided.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
2. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; ACL: Minneapolis, MN, USA, 2019; Volume 1, pp. 4171–4186. [CrossRef]
3. Shamshad, F.; Khan, S.; Zamir, S.W.; Khan, M.H.; Hayat, M.; Khan, F.S.; Fu, H. Transformers in medical imaging: A survey. *Med. Image Anal.* **2023**, *88*, 102802. [CrossRef]
4. Yang, X.; Bian, J.; Hogan, W.R.; Wu, Y. Clinical concept extraction using transformers. *J. Am. Med. Inform. Assoc.* **2020**, *27*, 1935–1942. [CrossRef]
5. Xiao, H.; Li, L.; Liu, Q.; Zhu, X.; Zhang, Q. Transformers in medical image segmentation: A review. *Biomed. Signal Process. Control* **2023**, *84*, 104791. [CrossRef]
6. Stylianou, N.; Vlahavas, I. TransforMED: End-to-End transformers for evidence-based medicine and argument mining in medical literature. *J. Biomed. Inform.* **2021**, *117*, 103767. [CrossRef]
7. Alicante, A.; Benerecetti, M.; Corazza, A.; Silvestri, S. A distributed architecture to integrate ontological knowledge into information extraction. *Int. J. Grid Util. Comput.* **2016**, *7*, 245–256. [CrossRef]
8. Yin, Y.; Lai, S.; Song, L.; Zhou, C.; Han, X.; Yao, J.; Su, J. An External Knowledge Enhanced Graph-based Neural Network for Sentence Ordering. *J. Artif. Intell. Res.* **2021**, *70*, 545–566. [CrossRef]
9. Gu, T.; Zhao, H.; He, Z.; Li, M.; Ying, D. Integrating external knowledge into aspect-based sentiment analysis using graph neural network. *Knowl.-Based Syst.* **2023**, *259*, 110025. [CrossRef]
10. Marcus, G. Deep Learning: A Critical Appraisal. *arXiv* **2018**, arXiv:abs/1801.00631.
11. Liu, J.; Chang, W.; Wu, Y.; Yang, Y. Deep Learning for Extreme Multi-label Text Classification. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; ACM: Tokyo, Japan, 2017; pp. 115–124. [CrossRef]
12. Gargiulo, F.; Silvestri, S.; Ciampi, M.; De Pietro, G. Deep neural network for hierarchical extreme multi-label text classification. *Appl. Soft Comput.* **2019**, *79*, 125–138. [CrossRef]
13. Nentidis, A.; Bougiatiotis, K.; Krithara, A.; Paliouras, G.; Kakadiaris, I. Results of the fifth edition of the BioASQ Challenge. In Proceedings of the BioNLP 2017 Workshop, Vancouver, BC, Canada, 4 August 2017; ACL: Vancouver, BC, Canada, 2017; pp. 48–57.
14. Nentidis, A.; Krithara, A.; Bougiatiotis, K.; Paliouras, G. Overview of BioASQ 8a and 8b: Results of the Eighth Edition of the BioASQ Tasks a and b. In Proceedings of the Working Notes of CLEF 2020—Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, 22–25 September 2020; CEUR Workshop Proceedings; Cappellato, L., Eickhoff, C., Ferro, N., Névéol, A., Eds.; CEUR-WS.org: Aachen, Germany, 2020; Volume 2696.
15. Nentidis, A.; Katsimpras, G.; Vandorou, E.; Krithara, A.; Paliouras, G. Overview of BioASQ Tasks 9a, 9b and Synergy in CLEF2021. In Proceedings of the Working Notes of CLEF 2021—Conference and Labs of the Evaluation Forum, Bucharest, Romania, 21–24 September 2021; CEUR Workshop Proceedings; Faggioli, G., Ferro, N., Joly, A., Maistro, M., Pirpi, F., Eds.; CEUR-WS.org: Aachen, Germany, 2021; Volume 2936.
16. Nentidis, A.; Krithara, A.; Paliouras, G.; Gasco, L.; Krallinger, M. BioASQ at CLEF2022: The Tenth Edition of the Large-scale Biomedical Semantic Indexing and Question Answering Challenge. In *Advances in Information Retrieval*; Hagen, M., Verberne, S., Macdonald, C., Seifert, C., Balog, K., Norvaag, K., Setty, V., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 429–435.
17. Peng, S.; You, R.; Wang, H.; Zhai, C.; Mamitsuka, H.; Zhu, S. DeepMeSH: Deep semantic representation for improving large-scale MeSH indexing. *Bioinformatics* **2016**, *32*, 70–79. [CrossRef] [PubMed]
18. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2019**, *36*, 1234–1240. [CrossRef] [PubMed]
19. You, R.; Liu, Y.; Mamitsuka, H.; Zhu, S. BERTMeSH: Deep contextual representation learning for large-scale high-performance MeSH indexing with full text. *Bioinformatics* **2020**, *37*, 684–692. [CrossRef] [PubMed]
20. Gu, Y.; Tinn, R.; Cheng, H.; Lucas, M.; Usuyama, N.; Liu, X.; Naumann, T.; Gao, J.; Poon, H. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Trans. Comput. Health* **2022**, *3*, 1–23. [CrossRef]
21. Mustafa, F.E.; Boutalbi, R.; Iurshina, A. Annotating PubMed Abstracts with MeSH Headings using Graph Neural Network. In Proceedings of the Fourth Workshop on Insights from Negative Results in NLP, Dubrovnik, Croatia, 5 May 2023; Association for Computational Linguistics: Dubrovnik, Croatia, 2023; pp. 75–81. [CrossRef]
22. Gargiulo, F.; Silvestri, S.; Ciampi, M. Exploit Hierarchical Label Knowledge for Deep Learning. In Proceedings of the 2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS), Córdoba, Spain, 5–7 June 2019; pp. 539–542. [CrossRef]

23. Brust, C.A.; Denzler, J. Integrating Domain Knowledge: Using Hierarchies to Improve Deep Classifiers. In *Pattern Recognition*; Palaiahnakote, S., Sanniti di Baja, G., Wang, L., Yan, W.Q., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 3–16.
24. Chen, F.; Yin, G.; Dong, Y.; Li, G.; Zhang, W. KHGCN: Knowledge-Enhanced Recommendation with Hierarchical Graph Capsule Network. *Entropy* **2023**, *25*, 697. [CrossRef] [PubMed]
25. Kowsari, K.; Brown, D.E.; Heidarysafa, M.; Meimandi, K.J.; Gerber, M.S.; Barnes, L.E. HDLTex: Hierarchical Deep Learning for Text Classification. In Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 18–21 December 2017; pp. 364–371. [CrossRef]
26. Mukherjee, A.; Garg, I.; Roy, K. Encoding Hierarchical Information in Neural Networks Helps in Subpopulation Shift. *IEEE Trans. Artif. Intell.* **2023**, 1–2. [CrossRef]
27. Jiang, Y.; Wang, Y. Topic-aware hierarchical multi-attention network for text classification. *Int. J. Mach. Learn. Cybern.* **2023**, *14*, 1863–1875. [CrossRef]
28. Aminizadeh, S.; Heidari, A.; Toumaj, S.; Darbandi, M.; Navimipour, N.J.; Rezaei, M.; Talebi, S.; Azad, P.; Unal, M. The applications of machine learning techniques in medical data processing based on distributed computing and the Internet of Things. *Comput. Methods Programs Biomed.* **2023**, *241*, 107745. [CrossRef]
29. Woźniak, M.; Wieczorek, M.; Siłka, J. BiLSTM deep neural network model for imbalanced medical data of IoT systems. *Future Gener. Comput. Syst.* **2023**, *141*, 489–499. [CrossRef]
30. Joloudari, J.H.; Marefat, A.; Nematollahi, M.A.; Oyelere, S.S.; Hussain, S. Effective Class-Imbalance Learning Based on SMOTE and Convolutional Neural Networks. *Appl. Sci.* **2023**, *13*, 4006. [CrossRef]
31. Xiong, J.; Yu, L.; Niu, X.; Leng, Y. XRR: Extreme multi-label text classification with candidate retrieving and deep ranking. *Inf. Sci.* **2023**, *622*, 115–132. [CrossRef]
32. Ye, X.; Xiao, M.; Ning, Z.; Dai, W.; Cui, W.; Du, Y.; Zhou, Y., NEEDED: Introducing Hierarchical Transformer to Eye Diseases Diagnosis. In Proceedings of the 2023 SIAM International Conference on Data Mining (SDM), Minneapolis, MN, USA, 27–29 April 2023; SIAM: Philadelphia, PA, USA, 2023; pp. 667–675. [CrossRef]
33. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, AZ, USA, 2–4 May 2013; Workshop Track Proceedings; Bengio, Y., LeCun, Y., Eds.; ICLR: Scottsdale, AZ, USA, 2013.
34. Silvestri, S.; Gargiulo, F.; Ciampi, M. Improving Biomedical Information Extraction with Word Embeddings Trained on Closed-Domain Corpora. In Proceedings of the 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, 29 June–3 July 2019; pp. 1129–1134. [CrossRef]
35. Gargiulo, F.; Silvestri, S.; Ciampi, M. A Big Data architecture for knowledge discovery in PubMed articles. In Proceedings of the 2017 IEEE Symposium on Computers and Communications, ISCC 2017, Heraklion, Greece, 3–6 July 2017; IEEE: Heraklion, Greece, 2017; pp. 82–87. [CrossRef]
36. Řehůřek, R.; Sojka, P. Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, La Valleta, Malta, 22 May 2010; ELRA: Luxemburg, 2010; pp. 45–50. [CrossRef]
37. Tsoumakas, G.; Katakis, I.; Vlahavas, I. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 667–685. [CrossRef]
38. Manning, C.D.; Raghavan, P.; Schütze, H. *Chetion to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2010. [CrossRef]
39. Aho, A.V.; Hopcroft, J.E.; Ullman, J.D. On finding lowest common ancestors in trees. *SIAM J. Comput.* **1976**, *5*, 115–132. [CrossRef]
40. Gargiulo, F.; Silvestri, S.; Ciampi, M. Deep Convolution Neural Network for Extreme Multi-label Text Classification. In Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2018)—Volume 5: HEALTHINF, Funchal, Madeira, Portugal, 19–21 January 2018; SciTePress: Madeira, Portugal, 2018; pp. 641–650. [CrossRef]
41. Zaman, S.; Moon, T.; Benson, T.; Jacobs, S.A.; Chiu, K.; Van Essen, B. Parallelizing Graph Neural Networks via Matrix Compaction for Edge-Conditioned Networks. In Proceedings of the 2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid), Taormina, Italy, 16–19 May 2022; pp. 386–395. [CrossRef]
42. Petit, Q.R.; Li, C.; Emad, N. Distributed and Parallel Sparse Computing for Very Large Graph Neural Networks. In Proceedings of the 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, 17–20 December 2022; pp. 6796–6798. [CrossRef]
43. Fu, Q.; Ji, Y.; Huang, H.H. TLPGNN: A Lightweight Two-Level Parallelism Paradigm for Graph Neural Network Computation on GPU. In Proceedings of the 31st International Symposium on High-Performance Parallel and Distributed Computing, Minneapolis, MN, USA, 27 June–1 July 2022; Association for Computing Machinery: New York, NY, USA, 2022; pp. 122–134. [CrossRef]
44. Buonaiuto, G.; Gargiulo, F.; De Pietro, G.; Esposito, M.; Pota, M. Best practices for portfolio optimization by quantum computing, experimented on real quantum devices. *Sci. Rep.* 2023, *in press*. [CrossRef] [PubMed]