

# Hierarchical Text Classification: a review of current research

Alessandro Zangari<sup>a,1</sup>, Matteo Marcuzzo<sup>a,\*</sup>, Michele Schiavinato<sup>a</sup>, Matteo Rizzo<sup>a</sup>,  
Andrea Gasparetto<sup>b</sup> and Andrea Albarelli<sup>a</sup>

<sup>a</sup>Ca' Foscari Department of Environmental Sciences, Informatics and Statistics, Via Torino 155, Mestre (VE), 30172, Italy

<sup>b</sup>Ca' Foscari Department of Management, Cannaregio 873, Fondamenta San Giobbe, Venice, 30121, Italy

---

## ARTICLE INFO

### Keywords:

Hierarchical Text Classification  
Multilabel Classification  
Hierarchical Metrics  
Natural Language Processing  
Text Classification Survey

## ABSTRACT

It is often the case that collections of documents are annotated with hierarchically-structured concepts. However, the benefits of this structure are rarely taken into account by commonly-used classification techniques. Conversely, Hierarchical Text Classification methods are devised to take advantage of the labels' organization to boost classification performance. With this work, we aim to deliver an updated overview of current research in this domain. We begin by defining the task and framing it within the broader text classification area, examining important shared concepts such as text representation. Then, we dive into details regarding the specific task, providing a high-level description of its traditional approaches. We then summarize recently proposed methods, highlighting their main contributions. We additionally provide statistics for the most adopted datasets and describe the benefits of using evaluation metrics tailored to hierarchical settings. Finally, a selection of recent proposals is benchmarked against non-hierarchical baselines on five domain-specific datasets.

---

## 1. Introduction

Text Classification (TC) is one of the most widely researched tasks within the Natural Language Processing (NLP) community (Gasparetto et al., 2022a). TC methods are supervised learning algorithms whose objective is to map documents (*i.e.*, pieces of text) to a predefined set of labels. Feature extraction techniques are applied to documents such as to produce numerical representations (text embeddings) of the text they are composed of. The ultimate goal is to maximize the discriminative power of such representations. In practice, the most common classification setting is *multiclass*, where only one label (*i.e.*, class) can be associated with each document. There might only be two labels to choose from (binary classification), as well as multiple. In contrast, *multilabel* classification allows every document to be labeled with multiple categories. Regardless, TC has many practical applications, some of the most common include topic labeling, sentiment analysis, and named entity recognition (Li et al., 2022).


Theoretically speaking, binary classification is the most generic classification scenario, as long as categories are stochastically independent (Sebastiani, 2002). If this is the case, the problem can be translated into  $|C|$  independent problems, where  $C$  is a set of classes. However, there are many practical scenarios in which this is not the case; Hierarchical Text Classification (HTC) is one of these. HTC is a sub-task of TC, as well as part of the wider Hierarchical Multilabel Classification (HMC). Vens et al. (2008) define HMC as classification where instances (i) may belong to multiple classes simultaneously, and (ii) are organized within a hierarchy. Thus, as labels do have a dependency on each other (as made explicit by the hierarchy), it is clear that the simplifying independence assumption cannot be made.

### 1.1. What is Hierarchical Text Classification?

The distinguishing property of HTC datasets is that their labels are organized in a multi-level hierarchy, where each label can be seen as a node able to have many possible children. In these types of situations, the hierarchical structure is key to the achievement of well-performing classification methods. It is often the case that real-world classification datasets contain a large number of categories organized into a class hierarchy or taxonomy. This arrangement of

---

\*Corresponding author

 [alessandro.zangari@unive.it](mailto:alessandro.zangari@unive.it) (A. Zangari); [matteo.marcuzzo@unive.it](mailto:matteo.marcuzzo@unive.it) (M. Marcuzzo); [michele.schiavinato@unive.it](mailto:michele.schiavinato@unive.it) (M. Schiavinato); [matteo.rizzo@unive.it](mailto:matteo.rizzo@unive.it) (M. Rizzo); [andrea.gasparetto@unive.it](mailto:andrea.gasparetto@unive.it) (A. Gasparetto); [albarelli@unive.it](mailto:albarelli@unive.it) (A. Albarelli)

ORCID(s): 0000-0002-3634-6607 (A. Zangari); 0000-0002-0451-4899 (M. Marcuzzo); 0000-0001-8943-6158 (M. Schiavinato); 0000-0002-9723-211X (M. Rizzo); 0000-0003-4986-0442 (A. Gasparetto); 0000-0002-3659-5099 (A. Albarelli)

<sup>1</sup>Authors contributed equally to this work.

information can sometimes be found even in corpora for which the hierarchy was not initially devised. For example, a set of documents categorized by their topics can usually be organized within large macro-areas which encompass subsets of related subjects (for instance, the “sports” macro category can contain both “tennis” and “football”). Indeed, hierarchies allow for intuitive modeling of what could instead be complex relationships among labels (such as the *is-a* and *part-of* relationships).

The availability of TC datasets that integrate hierarchical structure in their labels, as well as a general interest in industrial applications that utilize TC, has led to a large number of new methods for HTC being proposed in recent years. The strength of these “hierarchical classifiers” comes from their ability to leverage the dependency between labels to boost their classification performance. This is particularly important when considering the wider task of multilabel TC, which, in general, can be quite difficult. This is especially true when dealing with large sets of labels that contain many similar or related labels (Manning et al., 2008a).

## 1.2. Related work

Text Classification is one of the most active research areas within NLP, and many surveys and reviews have been published in recent years. These cover a wide range of techniques used in NLP, from traditional methods to the latest deep learning applications (Li et al., 2022; Minaee et al., 2021; Kowsari et al., 2019; Gasparetto et al., 2022a,b). However, these works cover the broader TC field and do not cover HTC specifically (or mention it very briefly). In the following paragraphs, we instead present some notable works within the field of HTC. We briefly touch on seminal works that describe this field, while also highlighting recently published reviews that perform an analysis of existing methods.

One of the first works to directly address HTC is that of Koller and Sahami (1997). The authors already highlight some of the most notable characteristics of HTC, such as the inadequacy of flat classifiers as opposed to their hierarchical counterparts (which we describe in Section 3.1) and the proliferation of topic hierarchies. Sun and Lim (2001) similarly address the difficulties tied to utilizing flat approaches in hierarchical settings, as well as discussing the issues related to standard performance metrics. As we will discuss in Section 3.3, classification metrics such as precision, recall and accuracy assume independence between categories, which might give a skewed representation of a classifier’s real performance (*e.g.*, performing a misclassification on a child not but not on its parent should be considered better than entirely incorrect classifications). In subsequent work, Sun et al. (2003) propose a specification language to describe hierarchical classification methods to facilitate the description and creation of HTC systems.

More recently, Silla and Freitas (2011) give a precise definition of hierarchical classification and propose a unifying framework to classify this task across different domains (*i.e.*, not limited to text). They provide a comprehensive overview of this research area, including a conceptual and empirical comparison between different hierarchical classification approaches, as well as some remarks on the usage of specialized evaluation metrics. Stein et al. (2019), on the other hand, address HTC directly, proposing an evaluation of traditional and neural models on a hierarchical task. In particular, the authors aim to gauge the efficacy of different word embedding strategies (which we outline in Section 2.1) with several methods for the specific HTC task. Several methods are tested, also comparing the effect of different text embedding techniques by evaluating their effect on both standard and specialized metrics. Similarly to other authors, they also advocate for the inadequacy of traditional “flat” classification metrics in hierarchical settings.

## 1.3. Contributions

In this work, we propose an analysis of the current research trends related to HTC, performing a systematic search of all papers that have been published in the last 4 years (*i.e.*, between 2019 and 2022). We deem this range effective to analyze recent works while also providing a way to limit the scope. We collect papers querying the keywords “*hierarchical text classification*”, “*hierarchical multilabel*” and “*multilevel classification*” in Google Scholar<sup>2</sup>, DBLP<sup>3</sup>, PapersWithCode<sup>4</sup> and Web of Science<sup>5</sup>.

Moreover, in our experimental section, we report the performance of a set of recent proposals, as well as several baselines, on five datasets. Three of these datasets are popularly utilized in the literature, while two of them are newly proposed versions of existing collections. The main contributions of this work can be summarized as follows:

- We provide an extensive review of the state of current research in regard to Hierarchical Text Classification;

<sup>2</sup><https://scholar.google.com>

<sup>3</sup><https://dblp.org>

<sup>4</sup><https://paperswithcode.com>

<sup>5</sup><https://webofscience.com>

- We explore the NLP background with regard to text representation and the various neural architectures being utilized in recent research;
- We analyze HTC specifically, providing an analysis of common approaches to this paradigm and its evaluation measures;
- We summarize a considerable number of recent proposals for HTC, spanning between 2019 and 2022. Among these, we dive deeper into the discussion of several methods and how they approach the task, with the ultimate purpose of testing their implementation in this work;
- In the experimental section, we test a set of baselines and recent proposals on five HTC datasets, three of which are popular benchmarks, and two of which are newly proposed versions of existing collections;
- We release our code<sup>6</sup> and dataset splits for public usage in research. Datasets are available on Zenodo (Zangari et al., 2022);
- Lastly, we summarize our results, consider current research challenges in the field and bring this work to a conclusion.

#### 1.4. Structure of the article

The rest of the article is organized as follows. Section 2 introduces the main NLP topics and recent advancements relevant to the latest proposals in HTC. Section 3 then dives into the specifics of HTC, describing the various types of approaches of the literature and hierarchical evaluation measures. Section 4 summarizes recently proposed methods between the years 2019 and 2022, analyzing in more detail a subset of them we wish to explore in the experimental section. This section also introduces the most popular datasets utilized in HTC research. The experimental part of this survey begins in Section 5, which outlines the datasets utilized and the methods being benchmarked, as well as a discussion of the results. The manuscript then moves towards its end in Section 6, which briefly explores current research challenges and research directions in the specific field of HTC, and draws its conclusions in Section 7.

## 2. NLP background

In this section, we provide an overview of the NLP subjects which lay the foundation of any HTC method. First off, we describe text representation and classification from a generic point of view, highlighting some of the most prominent approaches to the extraction of features from text. Then, in order to provide sufficient background for recent methods discussed throughout this article, we highlight some of the most notable neural architectures being utilized in the literature as of now. Indeed, most TC approaches are based on such neural architectures.

### 2.1. Text Representation and Classification

The interpretation of text in a numerical format is the fundamental first step of any application that processes natural language. How text is viewed and represented has changed drastically in the last few decades, moving from relatively simpler statistics-based word counts to more semantically and syntactically meaningful vectorized representations (Jones, 1972; Pennington et al., 2014; Vaswani et al., 2017). A richer text representation translates into more meaningful features, which — if utilized adequately — lead to massive improvements in downstream task performance. In this section, we overview the main approaches to text representation, highlighting major milestones and how they differ from one another.

#### 2.1.1. Text segmentation

First off, bodies of text must be segmented into meaningful units — a process called *tokenization* for historical reasons (Manning et al., 2008b). Naturally, the most intuitive one is that of *words*, though they are far from the only atomic unit of choice for this task. Indeed, this is a non-trivial task because of many factors. For instance, vocabularies (*i.e.*, the set of unique words in a corpus of documents) might be too large and sparse when segmenting for words. Moreover, some languages do not have explicit word boundary markers (such as spaces for English).

This is a vast and interesting topic of its own, and we point readers to the work by Mielke et al. (2021) for further information on it. Very briefly, it is important to mention that recent approaches have begun to rely on

<sup>6</sup><https://gitlab.com/distratation/dsi-nlp-publib/-/tree/main/htc-survey-22>

*sub-word* tokenization approaches, which broadly operate on the assumption that common words should be kept in the vocabulary, while rarer words should be split into “sub-word” tokens. This allows for smaller and more dense vocabularies and has been successfully applied to many recent approaches. Examples of sub-word tokenization approaches include Byte-pair Encoding (Sennrich et al., 2016 Aug 7–12), WordPiece (Schuster and Nakajima, 2012 Mar 25–30), and SentencePiece (Kudo and Richardson, 2018 Oct 31 – Nov 4). Even more recently, some authors have argued for different forms of decomposition, such as ones utilizing underlying bytes (Graves, 2013), or even visual modeling based on the graphical representation of text (Salesky et al., 2021 Nov).

### 2.1.2. *Weighted word counts*

Some of the more traditional and widely utilized approaches in the past are based on word occurrence statistics, effectively ignoring sentence structure and word semantics entirely. The most common example is that of the *Bag-Of-Words* (BoW) representation; in it, documents are simply represented as a count of their composing words, which are then often weighted with normalizing terms such as the well-studied Term Frequency-Inverse Document Frequency (TF-IDF) (Jones, 1972). Briefly, TF-IDF measures word relevancy by weighting it positively by how frequently the word appears in a document (TF), but also negatively by how frequently it appears in all other documents (IDF). This way, words that frequently appear in a document are highlighted, but only if they do not appear often in other documents as well (as this negates their discriminative power).

### 2.1.3. *Word embeddings*

Weighted word counts such as TF-IDF weighted BoW fail to capture any type of semantic and syntactic property of text. A major milestone towards the development of more meaningful representations is the development of *word embeddings*, initially popularized by works such as Word2Vec (Mikolov et al., 2013a,b) and GloVe (Pennington et al., 2014). These vectorial representations of text are learned through unsupervised language modeling tasks; briefly, language modeling refers to the creation of statistical models created through word prediction tasks, and has been studied for many decades (Jurafsky and Martin, 2020). The resulting *language models* (LMs) are useful in a variety of tasks, one of which is indeed the extraction of meaningful word and sentence representations.

The LMs utilized to develop word embeddings are based on shallow neural networks pre-trained on massive corpora of documents, allowing them to develop meaningful vectorial representations for words. The underlying semantic properties of these representations have often been exemplified through vector arithmetic operations, such as the classic example of  $\vec{k}ing - \vec{m}an + \vec{w}oman \approx \vec{q}ueen$ . Nonetheless, at a practical level, these vectors — *i.e.*, word embeddings — can then be utilized as input features for downstream algorithms (*e.g.*, classifiers), leading to vast improvements in terms of performance.

### 2.1.4. *Contextualized Language models*

Word embeddings have sometimes been defined as “static”, as their early iterations produced representations that were unable to disambiguate the meaning of a word with multiple meanings (*i.e.*, polysemous words). An embedding for such a word, then, would be an average of its multiple meanings, leading to an inevitable loss of information. While much research has been dedicated to the addition of context to word embeddings (with notable results such as ELMO (Peters et al., 2018 Jun 1–6)), the introduction of the Transformer architecture (Vaswani et al., 2017) (see Section 2.2.3) has been certainly the most pivotal moment towards the development of contextualized LMs.

In layman’s terms, these models contextualize word embeddings by studying their surrounding words in a sentence (the “context”) (Devlin et al., 2019; Liu et al., 2019c; Radford et al., 2018, 2019; Brown et al., 2020). A notable change of Transformer-based LMs over previous approaches is the lack of recurrence in their architectures, which instead utilize the attention mechanism (Bahdanau et al., 2015; Luong et al., 2015) as their main component. This was a drastic change when considering that Recurrent Neural Networks (RNNs) (Rumelhart et al., Jan 1986; Sutskever et al., 2014 Dec 8–13) were the go-to approach for text interpretation before the introduction of purely attention-based models, as they are particularly effective when dealing with sequential data. However, Transformers are much more parallelizable, and also have been shown to scale positively with increased network depth, something that is not true for RNNs (Kaplan et al., 2020). Therefore, larger and larger Transformer-based LMs can be built (both in terms of training data and model parameters), a practice that has seen widespread use in the most recent literature. Some models have begun to be referred to with the moniker of *Large Language Models* (LLMs) because of the massive amount of parameters they contain. Examples of this trend include GPT-3 (175 billion parameters) (Brown et al., 2020), GShard (600 billion parameters) (Lepikhin et al., 2021 May 4), and Switch-C (1.6 trillion parameters) (Fedus et al., 2022).

### 2.1.5. Classification

In terms of classification, traditional word representation methods such as TF-IDF (and, to a lesser extent, word embeddings) have been widely utilized as input features for traditional classification methods such as Decision Trees (Safavian and Landgrebe, 1991; Ho, 1995), Support Vector Machines (Cortes and Vapnik, 1995; Boser et al., 1992), and probabilistic graphical models (*e.g.*, Naive Bayes and Hidden Markov Models) (Xu et al., 2017; van den Bosch, 2017). The same can be said about word embeddings, which have however seen much greater use with specialized neural network architectures such as Convolutional Neural Networks (CNNs) (Zhang and Wallace, 2017; Kim, 2014) and RNNs (Liu et al., 2016; Kowsari et al., 2017). Transformer-based LMs such as the Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) and Generative Pre-trained Transformer (GPT) (Radford et al., 2018), on the other hand, have showcased outstanding classification results by passing the contextualized embeddings through a simple feed-forward layer. The ease of adaptation of these models has highlighted the importance of developing well-crafted representations for text. For a more in-depth description of classification approaches in NLP, we refer the readers to Gasparotto et al. (2022a).

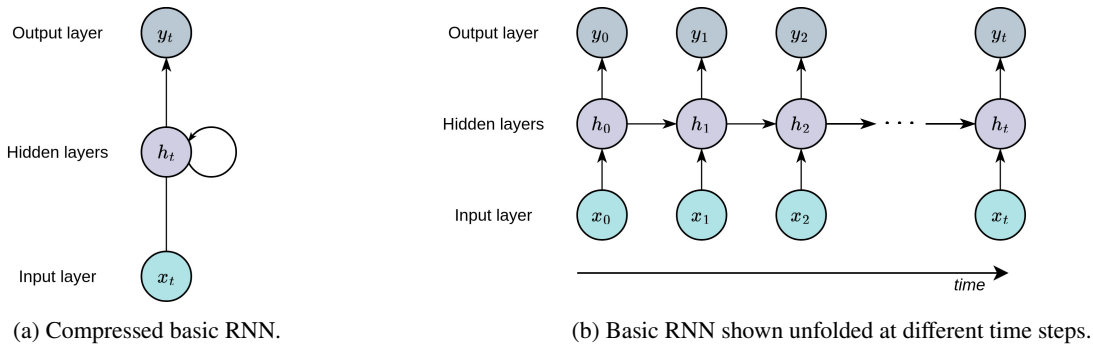
## 2.2. Notable neural architectures

Much of the progress achieved in the development of meaningful text representation is attributable to neural networks, and deep learning in particular. While earlier approaches discussed (such as Word2Vec and GloVe) were initially based on shallow Multilayer Perceptrons (Mikolov et al., 2013a; Pennington et al., 2014), better results were later obtained with larger and deeper networks (*i.e.*, more layers, more parameters). In this section, we briefly overview some of the most influential neural architectures and mention notable applications in the field of text representation.

### 2.2.1. Recurrent Neural Networks

RNNs (Rumelhart et al., Jan 1986) are networks particularly well suited to environments that utilize sequential data, such as text or time series. This particular architecture allows these networks to retain a certain amount of “memory”, making them able to extract latent relationships between elements within sequences. In practice, this is done by utilizing information from prior inputs to influence the current input and output of the network.

Simple RNNs for text processing are commonly fed a sequence of word embeddings, which are processed sequentially. At each time step, the network receives both the next word vector as well as the hidden state of the previous time step (Fig. 1). Because of their structure, standard RNN architectures are vulnerable to gradient-related issues, such as vanishing and exploding gradients (Pascanu et al., 2013). In response to this issue, these architectures are frequently enhanced with gating mechanisms, the most popular being Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014a). Briefly, these gates allow the network to control which and how much information to retain, such as to enable the modeling of long-term dependencies. RNNs that utilize these gates are often referred to with the acronym of the gate itself, *i.e.*, LSTM networks, and GRU networks.



**Figure 1:** Exemplification of a simple RNN structure.

A simple RNN (Elman, 1990) can be defined as in the equation below, where  $h_t$  and  $x_t$  are the hidden state and input vector at time  $t$  respectively, while  $\mathbf{W}_h$  and  $\mathbf{W}_x$  are learnable weight matrices:

$$h_t = \sigma(\mathbf{W}_h h_{t-1} + \mathbf{W}_x x_t + \mathbf{b}_h) \quad (1)$$



Here,  $\sigma$  is an activation function, typically tanh or Rectified Linear Unit (ReLU), and  $\mathbf{b}$  is a bias term. The output  $y_t$  at each time step is usually derived from the state of the last layer of the RNN at all time steps, for instance:

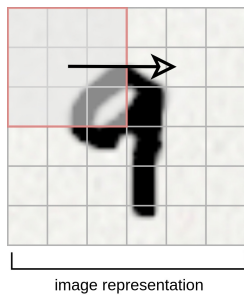
$$y_t = \sigma(\mathbf{W}_y h_t + \mathbf{b}_y) \quad (2)$$

In the context of text representation, RNNs based on encoder-decoder architectures have been widely utilized to extract meaningful textual representations (Cho et al., 2014b). Briefly, an encoder-decoder structure can be understood as an architecture by which inputs are mapped to a compressed yet meaningful representation (contained in the hidden states between encoder and decoder). This representation should hopefully capture the most relevant features of the input, and can then be decoded for a variety of different tasks (e.g., translation). Autoencoders (Kramer, 1991) are a particular class of encoder-decoder networks that attempt to regenerate the input exactly; they are particularly useful in creating efficient representations of unlabeled data.

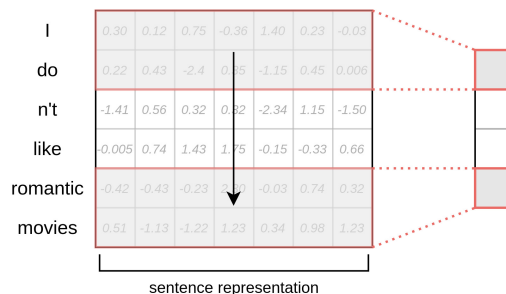
In this context, the hidden states between the encoder and the decoder make for a semantically meaningful and compact representation of input words. The introduction of bidirectionality (influence in both left-to-right and right-to-left directions) in RNNs has also been proved to be beneficial and has been used to achieve notable results such as the aforementioned ELMo (Peters et al., 2018 Jun 1–6), a bidirectional-LSTM based language model which marked one of the first milestones towards the development of contextualized word embeddings. Nonetheless, RNNs have inherent limitations because of how they process data sequentially, making them ill-suited for parallelization. Furthermore, despite the improvements introduced by LSTM and GRUs, RNNs still struggle with long sequences because of memory constraints and their tendency to forget earlier parts of the sequence (Pascanu et al., 2013).

### 2.2.2. Convolutional Neural Networks

CNNs (LeCun et al., 1989) are well-known neural architectures, originally devised for Computer Vision (CV) applications. However, these networks have since been extended to other fields, achieving excellent results in NLP tasks as well (Kim, 2014). The core structural element of CNNs is the *convolutional layer*, which applies a feature detector (*filter* or *kernel*) across subsets of the input (i.e., the convolution operation) to extract features. While this has a more intuitive interpretation in CV (where the filter moves across the image to search for features), the same reasoning can be applied to text. Intuitively, convolution as applied to images can be thought of as a weighted average of each pixel based on its neighborhood; the general idea of the process is outlined in Fig. 2a. If we consider a vectorial representation of text (i.e., word embeddings), applying a filter as wide as the embedding size (a common approach) allows us to search for features within the sentence, as shown in Fig. 2b. CNNs often use pooling operators (such as max or average) to reduce the size of the learned feature maps, as well as to summarize information.



(a) A convolutional filter (top left) sliding across a digit from the MNIST dataset (Deng, 2012).



(b) A convolutional filter sliding across the vectorial representation of a sentence. Here, the filter is sliding in the conventional reading direction.

**Figure 2:** Exemplification of the application of convolutional filters in images and text.

Various works have tested the efficacy of CNNs, especially as feature extractors on word embeddings (Kim, 2014). An obvious upside of these architectures is their speed (as they are much more parallelizable than RNNs). Thus, CNNs produce efficient yet effective latent representations that can be used to solve a variety of tasks (e.g., classification). Recent works have also revitalized the interest of CNNs in NLP by introducing Temporal CNNs. In short, these aim to extend CNNs by allowing them to capture high-level temporal information (Lea et al., 2017; Yan et al., 2020).

### 2.2.3. Transformer Networks and the Attention mechanism

As previously mentioned, one of the most influential neural architectures introduced in recent years — especially in terms of text processing — is the Transformer architecture (Vaswani et al., 2017). The foundational framework is that of an encoder-decoder with a variable number of encoder and decoder blocks. In Section 2.1.4, we briefly touched upon the main innovation introduced by Transformers, which is the complete lack of recurrence as a learning mechanism. Instead, Transformers model context dependency between words entirely through the *attention mechanism* (Bahdanau et al., 2015; Luong et al., 2015), which is outlined in the remaining part of this section.

Attention had been utilized before as an extension to various architectures, with the principal aim of weighing the contributions of different parts of the input differently (*i.e.*, “pay attention” to them). In essence, it is a weighting strategy, devised to learn how different components contribute to a result. It was initially proposed in the machine translation domain (Bahdanau et al., 2015) as an alignment mechanism that matched each word of the output (translated sentence) to the respective words in the input sequence (original sentence). The rationale behind this was that, when translating a sentence, a good translation can only be obtained by looking at the context of words and paying attention to specific words.

Vaswani et al. (2017) used this mechanism in the Transformer architecture to allow the model to process all input tokens simultaneously, rather than sequentially, as was the case in previous recurrent networks. Input sequences are fed to the Transformer encoder at the same time, and a *positional encoding* scheme is used in the first layers of the encoder and decoder to inject some ordering information in the word embeddings. This ensures word ordering properties are not lost, *e.g.*, that two words appearing in different positions in a sentence will have a different representation. Then, in all the remaining layers, Transformers use self-attention layers to learn dependencies between tokens. The layers are “self”-attentive as each token pays attention to every other token in a sentence, which is the main learning mechanism that allows for the disposal of recurrence. Indeed, since tokens in the input sequence are being processed simultaneously, the encoders are able to look at surrounding tokens in the same sentence and to produce context-dependent token representations (Gasparetto et al., 2022a).

Stacking several self-attention layers produces a multi-head attention (MHA) layer, whose structure is shown in Figure 3. Vaswani et al. (2017) argue that having multiple attention heads in the layer enables the model to pay attention to different information in distinct feature spaces, and at different positions. Input sequences in the attention heads are transformed using linear transformations to generate three different representations, which the authors name  $\mathbf{Q}$  (queries),  $\mathbf{K}$  (keys), and  $\mathbf{V}$  (values), following the naming convention used in information retrieval (as in Eq. 3):

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_{\mathbf{Q}}, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_{\mathbf{K}}, \quad \mathbf{V} = \mathbf{W}_{\mathbf{V}} \quad (3)$$

Where  $\mathbf{W}_{\mathbf{Q}}, \mathbf{W}_{\mathbf{K}} \in \mathbb{R}^{dim \times d_k}$ ,  $\mathbf{W}_{\mathbf{V}} \in \mathbb{R}^{dim \times d_v}$  are the learned weight matrices. In the Transformer architecture,  $\mathbf{K}$  and  $\mathbf{V}$  are always generated from the same sequence (as we will discuss,  $\mathbf{Q}$  is used differently in the encoder and decoder part). The transformed sequences are then used to compute the scaled dot-product attention, as follows:

$$\mathbf{Z}_k = \text{ScaledDotAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}} \right) \mathbf{V} \quad (4)$$

While many definitions of attention exist in the literature, the authors decided to use the scaled dot-product mainly for efficiency reasons. The product of the key and query matrices is scaled by  $\sqrt{d_k}$  to improve the stability of the gradient computation. At an intuitive level, we may envision a word  $q \in \mathbf{Q}$  as being queried for similarity/relatedness against keys  $k \in \mathbf{K}$ , finally obtaining the relevant word representation by multiplying by  $\mathbf{V}$ .

The final result of the MHA layer is the concatenation of all heads multiplied by matrix  $\mathbf{W}_{\mathbf{O}} \in \mathbb{R}^{hd_v \times dim}$  that reduces the output to the desired dimension  $\mathbb{R}^{N \times dim}$ :

$$\mathbf{Z} = \text{Concat}(\mathbf{Z}_k)\mathbf{W}_{\mathbf{O}} \quad (5)$$

In encoder blocks,  $\mathbf{Q}$ ,  $\mathbf{K}$  and  $\mathbf{V}$  are all generated by the input sequence and have the same size; the general structure of an encoder block can be seen in Fig. 4a. In decoder blocks,  $\mathbf{Q}$  comes from the previous decoder layer, while  $\mathbf{K}$  and  $\mathbf{V}$  come from the output of the associated encoder. These blocks structurally differ from encoders in the presence of another MHA layer, which precedes the standard one and is introduced to mask future tokens in a sentence (Fig. 4b). This allows the decoder to be “autoregressive” during training, as it otherwise would trivially look forward in a sentence to obtain the result.

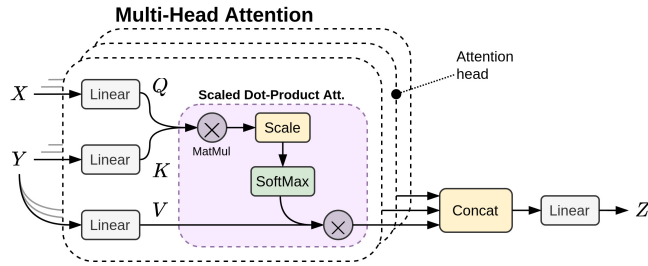
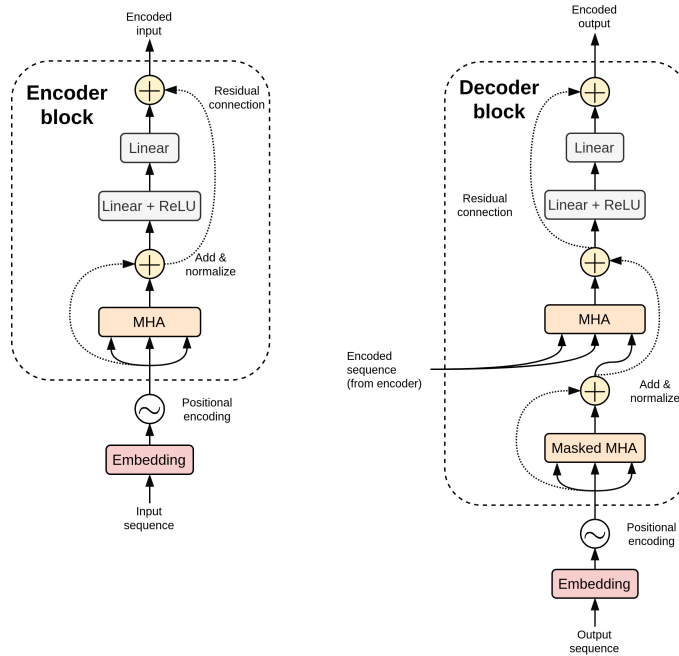


Figure 3: The multi-head attention layer used in the Transformer architecture.



(a) Encoder block of a Transformer. (b) Decoder block of a Transformer.

Figure 4: Transformer encoder-decoder architecture.

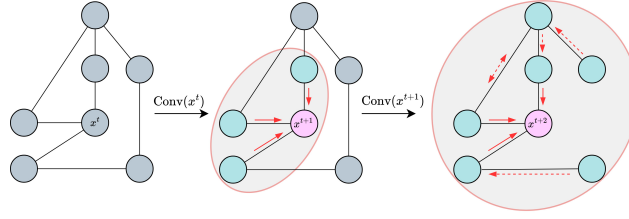
With the widespread usage of Transformer blocks with MHA, researchers have studied the feature extracted at each layer. Interestingly, some of these works found that each encoding block may focus on the extraction of linguistic features at different levels: syntactic features are mostly extracted in the first blocks, while deeper layers progressively focus on semantic features (de Vries et al., 2020; Jawahar et al., 2019). This “layer specialization” phenomenon suggests that stacking attention layers has the effect of creating expressive representations that blend together morphological and grammatical features.

### 2.2.4. Graph Neural Networks

The ubiquity of graph structures in most domains has sparked much interest in the application of neural networks directly to graph representations. In the NLP domain, a body of text can be represented as a graph of words, where connections represent relations that are potentially semantic or grammatical in nature. As a simple example, a sentence could be represented by word nodes, and adjacent words in the sentence would be linked by an edge. Entire documents can also be linked together in a graph, for instance in citation networks, or simply considering their relatedness (Zhou et al., 2020a; Gasparetto et al., 2022a).

In the particular context of hierarchical classification, graphs of labels have often been used to propagate hierarchy information between connected labels, with connections usually representing parent-child relations.





**Figure 5:** The propagation effect operated by two sequential graph convolutions with respect to node  $x$ . The red arrows showcase the information flow toward the target node.

*Message Passing* The principle behind graph processing models is generalized by the Message Passing Neural Network (MPNN) (Gilmer et al., 2017). In this model, a *message passing phase*, lasting  $T$  time steps, is used to update nodes and edges representations by propagating information along edges.

First, for a graph  $G = (V, E)$ , the message at time  $t + 1$  is computed for each node  $u \in V$ , depending on the previous values of the nodes and edges  $e \in E$ . Therefore, for a node  $u$ :

$$m_u^{t+1} = \sum_{v \in \mathcal{N}(u)} \phi(x_u^t, x_v^t, e_{uv}^t) \quad (6)$$

where  $\phi$  is a function learned by a neural network, function  $\mathcal{N}(u)$  gives the neighbor nodes of  $u$ , and  $x \in X$  are node embeddings (representations). Eq. 6 uses the sum operation to aggregate the multiple messages coming from the neighbors of  $u$ , although it could be replaced with other permutation-invariant operations, such as the average or minimum. Once the messages have been computed, the node embeddings are updated using an update function  $\sigma$ , which is also learned by a neural network:

$$x_u^{t+1} = \sigma(x_u^t, m_u^{t+1}) \quad (7)$$

Analogously, Eq. 6 and 7 could be adapted to compute the message starting from neighbor edges instead of nodes and to update the edge representation accordingly. Finally, in the *readout phase* values from all nodes are aggregated by summing them together and the output is used for a graph-level classification task.

*Graph Convolution* The concept of message passing lends itself to the definition of a *convolution operation* for graphs that can be used within Graph Convolutional Neural Networks (GCNs) (Yao et al., 2019). In the literature, two main categories of GCNs are typically distinguished: spatial-based and spectral-based. Similarly to the conventional convolution operation over an image, spatial-based GCNs define graph convolutions based on the graph topology, while spectral-based methods are based on the graph's spectral representation (Zhou et al., 2020a; Shuman et al., 2013; Wu et al., 2021). In this paragraph, we will only discuss the former type, which is more closely related to the message-passing concept.

While many different definitions of convolution have been proposed, a simple spatial convolution operator on a graph can be defined as:

$$\text{Conv}(x_u^t) = \sigma \left( \bigodot_{v \in \mathcal{N}(u)} \phi(x_u^t, x_v^t) \right) \quad (8)$$

in which  $\bigodot$  is a permutation invariant operation. The result of this operation can be used to update each node representation, as exemplified in Fig. 5. When  $\bigodot = \sum$  and  $\phi(x_u^t, x_v^t) = x_v^t \mathbf{W}^T + b$ , which is a simple linear transformation of the representation of neighbor nodes, this operation can be defined in matrix form as:

$$\text{Conv}(\mathbf{X}) = \sigma(\mathbf{A}(\mathbf{X}\mathbf{W}^T + \mathbf{b})) \quad (9)$$

In the equation above,  $\mathbf{A} \in \mathbb{R}^{n \times n}$  is the adjacency matrix,  $\mathbf{X} \in \mathbb{R}^{n \times d}$  contain the nodes' representation of dimensionality  $d$ , and  $\mathbf{W} \in \mathbb{R}^{d_{out} \times d}$  and  $\mathbf{b} \in \mathbb{R}^{d_{out}}$  are the weight matrix and bias term respectively. In the above

operation, the multiplication by the adjacency matrix  $\mathbf{A}$  guarantees that only neighbor nodes contribute to the updated representation  $\mathbf{X}$ . Consequently, if multiple convolutions are stacked, the message from each node can propagate further in the graph. If too many layers are used, large portions of the graph could end up having similar nodes' representations, an issue often referred to as *oversmoothing* (Li et al., 2018 Feb).

As a natural extension of the operation defined in Eq. 9,  $\mathbf{A}$  can be weighted to reflect the importance of neighbors, for instance by multiplying each entry with edge weights. The attention mechanism can also be used to autonomously learn how much each node should contribute to their neighbors' representations. An example of this is Graph Attention Networks (GATs) (Veličković et al., 2018), which use the Transformer's MHA to compute the hidden states of each node.

### 2.2.5. Capsule Networks

Much recent literature explores the usages of Capsule Networks (CNs) (Hinton et al., 2011) to find structure in complex feature spaces. These networks group perceptrons — the base units of feed-forward networks — into *capsules*, which can essentially be interpreted as groups of standard neurons. Each capsule specializes in the identification of a specific type of entity, like an object part, or, in general, a concept. Since a capsule is a group of neurons, its output is a vector instead of a scalar, and its length represents the probability that an entity exists in the given input, as well as its spatial features. Capsules have been first applied to object recognition (Xi et al., 2017) to address the shortcomings of CNNs, such as the lack of rotational invariance. However, capsules have been proposed in NLP applications as well; for instance, TC datasets often present labels that can be grouped into related meta-classes that shares common concepts. A hierarchy of topics may include a “Sports” macro-topic, with “Swimming”, “Football”, and “Rugby” as sub-topics. However, the latter two sports are more similar, both being “team sports” and “ball sports”, something that is not made explicit by the hierarchy. CNs have been applied to the HTC task for their ability in modeling latent concepts and hence capture the latent structure present in the label space. It is expected that a better understanding of the labels' organization, such as modeling the “team sports” and “ball sports” concepts, can be effectively exploited to improve decision-making during classification (Wang et al., 2021b; Aly et al., 2019).

*Capsules* As briefly mentioned above, a capsule is a unit specialized in the following tasks:

- Recognizing the presence of a single entity (*e.g.*, deciding how likely it is that an object, or piece-of, is present in an image);
- Computing a vector that describes the instantiation parameters of the entity (*e.g.*, the spatial orientation of an object in an image, or local word ordering and semantics of a piece of text (Aly et al., 2019)).

As a result, capsules can specialize and estimate parameters about the selected entities. In contrast, neurons in standard MLPs can only output scalar values that cannot encapsulate such a richness of information. Within CNs, capsules are organized in layers, and the output of each child capsule is fed to all capsules in the next layer (*i.e.*, parent capsules) using weighted connections that depend on the level of agreement between child capsules. Higher layers tend to specialize in recognizing more high-level entities, estimating their probability using the information about sub-parts that are propagated by lower-level capsules (Sabour et al., 2017). Output between capsules is routed through the *dynamic routing* mechanism.

*Dynamic routing* As mentioned, each layer of capsules specializes in the recognition of specific entities; subsequent layers should make sense of the information extracted by previous layers and use it to recognize higher-level entities. For instance, some capsules may identify “human hands” in an image, and higher-level capsules could use this information to determine the presence of human bodies. However, not all capsules may have recognized entities that are explainable by looking at the more complex entities that are detected in higher layers. Continuing with the previous example, the presence of “hands” may be unlikely if no human body is detected by the following layer. The general principle behind the routing-by-agreement algorithm known as Dynamic Routing (Sabour et al., 2017) is that connections between capsules in layer  $l$  to a capsule  $j$  in layer  $l + 1$  should be weighted depending on how much capsules in  $l$  collectively agree on the output of capsule  $j$ . When many of them agree, it means that all the entities they recognized can be part of the composite entity recognized by capsule  $j$ , and as such their output should be sent mostly to capsule  $j$ .

The routing algorithm updates routing weights (*i.e.*, connections between capsules in different layers) so that they reflect the agreement between them. Let  $\{\mathcal{K}, \mathcal{L}\}$  be two subsequent layers of a CN, each made up of several capsules.

Predictions made by capsule  $i \in \mathcal{K}$  about the output of capsule  $j \in \mathcal{L}$  is computed as:

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i \quad (10)$$

Where  $\mathbf{u}_i$  is the activation vector of capsule  $i$  and matrix  $\mathbf{W}$  is the learned transformation matrix that encodes the part-to-whole relationship between pairs of capsules in two subsequent layers. A distinct weight matrix  $\mathbf{B}$  is used to store the connection weights  $\mathbf{B}_{ij}$  between each capsule  $i \in \mathcal{K}$  and capsule  $j \in \mathcal{L}$ . The agreement is measured using the dot product, and weights are updated as in the equation below:

$$\mathbf{B}_{ij} \leftarrow \mathbf{B}_{ij} + \hat{\mathbf{u}}_{j|i} \mathbf{v}_j \quad (11)$$

Where output  $\mathbf{v}_j$  represents the probability that the entity recognized by the capsule  $j$  is present in the current input and therefore should be consistent with the information extracted by lower-level capsules (*i.e.*, the “guesses”), which is encoded in vector  $\hat{\mathbf{u}}$ . Hence, the output of a capsule  $j \in \mathcal{L}$  is the weighted sum of the predictions made by all capsules in the previous layer  $\mathcal{K}$ . Note that, Since the activation vector must represent a probability, the *squash* function is used to shrink the output into the (0, 1) range:

$$\mathbf{v}_j \leftarrow \text{squash}(s_j) \quad s_j = \sum_{i \in \mathcal{K}} \mathbf{C}_{ij} \hat{\mathbf{u}}_{j|i} \quad (12)$$

During each iteration the *coupling coefficients* (or *routing weights*) are computed as follows:

$$\mathbf{C}_i \leftarrow \text{softmax}(\mathbf{B}_i), \quad \forall \text{ capsule } i \in \mathcal{K} \quad (13)$$

This process ensures that capsules in  $\mathcal{K}$  that were more in agreement with capsules in  $\mathcal{L}$  will send a stronger signal than capsules that made a different prediction, as opposed to higher-level capsules. Moreover, as Eq. 12 shows, the output of capsules in subsequent layers is dependent on the prediction made in previous layers. This means that the higher the number of capsules that agree on the most likely prediction of some capsules in the next layer (*i.e.*  $\hat{\mathbf{u}}$ ), the more they can influence the output of those capsules.

The routing process is repeated a fixed number of times before the algorithm proceeds to the next layer and stops when all the connections between capsules are weighted. Finally, this routing mechanism has been recently improved using the Expectation-Maximization algorithm (Hinton et al., 2018), to overcome some of the limitations of the former approach.

### 3. Hierarchical Text Classification

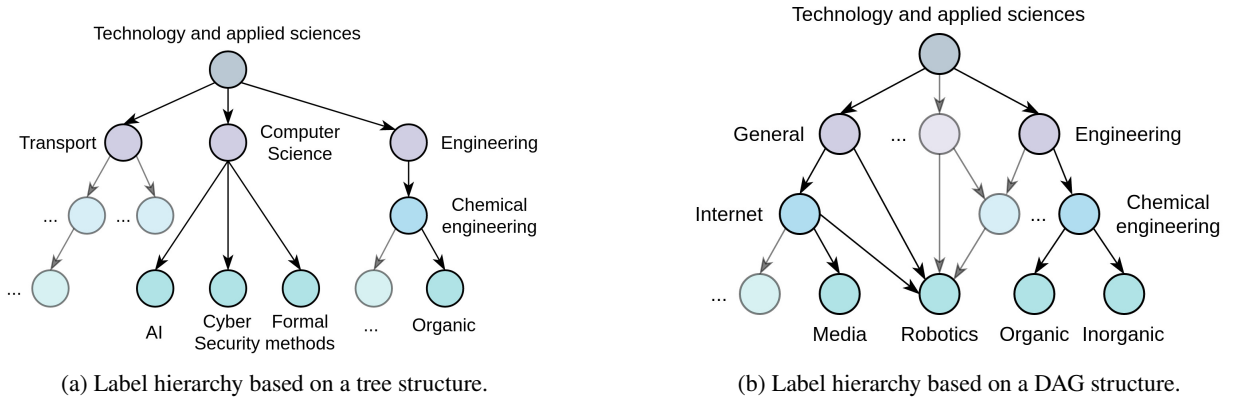
As a sub-task of the broader TC area, HTC methods have much in common with standard text classifiers. In this section, we will outline the main aspects which differentiate HTC from standard classification approaches, and how they can be leveraged to achieve a higher classification accuracy in the presence of a taxonomy of labels.

#### 3.1. Types of Hierarchical Classification

Standard classifiers focus on what HTC literature defines as *flat classification*. In it, categories are treated in isolation (*i.e.*, as having no relationship between one another) (Sun and Lim, 2001; Sun et al., 2003). In contrast, HTC deals with documents whose labels are organized in structures that resemble a tree or a directed acyclic graph (DAG) (Wang et al., 2022a; Peng et al., 2018). In these structures, each node contains a label to be assigned, such as in Fig. 6. Methods able to work with both trees and DAGs can be devised, though a simpler technique is to simply “unroll” or “flatten” sub-nodes with multiple parents for DAGs, thus obtaining a tree-like representation. For this reason, this article (and much of HTC literature) focuses on hierarchies with a tree structure.

HTC approaches can be divided into two groups: *local* and *global* approaches (Yu et al., 2022). Local approaches (sometimes called “top-down”) are defined as such because they “dissect” the hierarchy, constructing multiple local classifiers that work with a subset of the node labels. While more informed than flat classifiers — which ignore the hierarchy — there is an inevitable loss of hierarchical information, as the aggregation of these classifiers tends to ignore the holistic structural information of the taxonomic hierarchy.

Depending on the chosen approach, the amount of information regarding the hierarchy can be partial or absent (Zhang et al., 2022). Local approaches (Fig. 7b, 7c, 7d) have been criticized because of their structural issues, the



**Figure 6:** Hierarchically structured labels (inspired by Wang et al. (2022a)). Both forms are frequent in practice, though labels organized in a DAG require adaptation of either the method or the structure.

most notable one being that they may easily propagate misclassifications (Punera and Ghosh, 2008; Cerri et al., 2011). Furthermore, these models are often large in terms of trainable parameters, and may easily face exposure bias because of the lack of holistic structural information (Zhou et al., 2020b). While we will discuss this in more detail later, this arises from the fact, at test time, lower-level classifiers usually use the prediction of previous classifiers, thus leading to a discrepancy to the training process (which is usually based on ground truths). Global methods aim to solve these shortcomings, as well as being frequently less computationally expensive (as there is a single classifier) (Silla and Freitas, 2011). While the definition of global classifiers is purposely generic, one might imagine as global any classification algorithm that directly takes into account the hierarchy (for instance, many standard algorithms have been modified to this end). It is also worth noting that global and local approaches might also be combined (Wehrmann et al., 2018; Mao et al., 2019). Fig. 7 showcases the main approaches to HTC, which we now briefly summarize.

### 3.1.1. Flattened classifiers

The flat classification approach (Fig. 7a) reduces the task to a multiclass (or multilabel) classification problem, therefore discarding hierarchical information entirely. Typically, only leaf nodes are considered, and the classification of higher levels of the hierarchy is inherited from parent nodes (Zhang et al., 2022; Aly et al., 2019).

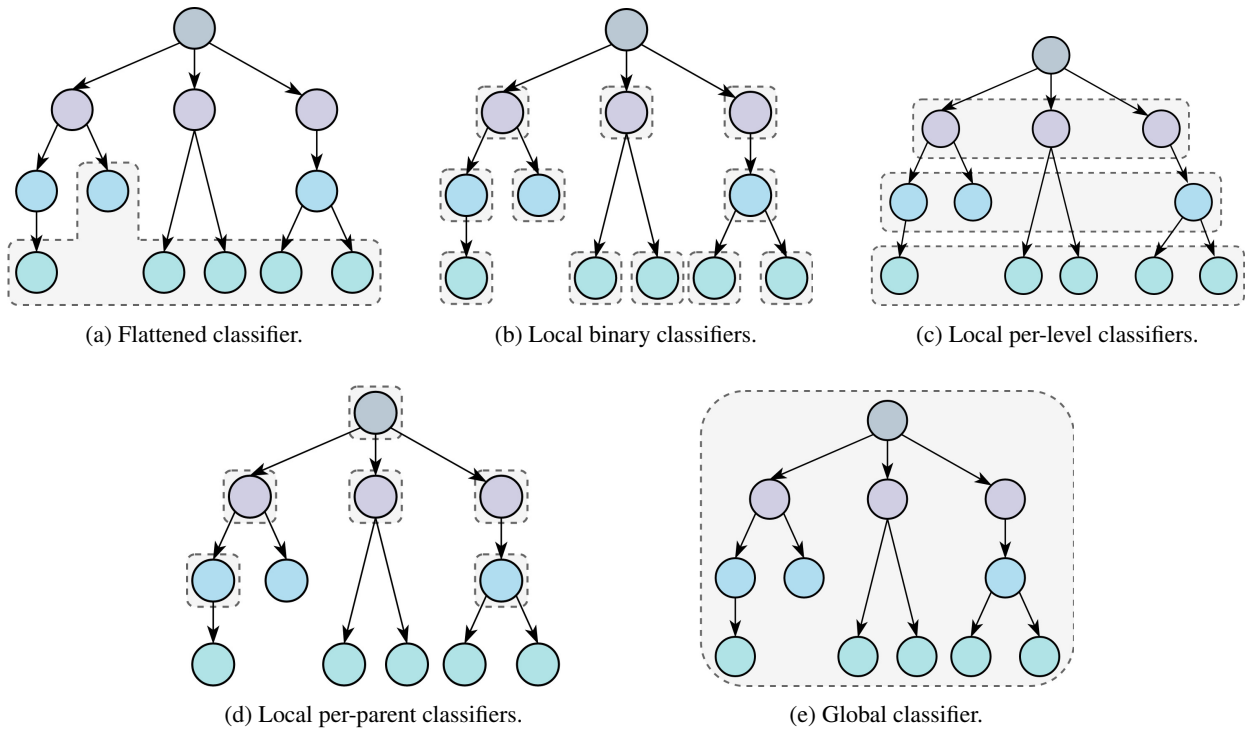
### 3.1.2. Local classifiers

Local classification approaches are generally divided into three categories which differ in how they dissect the hierarchy.

First off, the class of *local per-node (binary)* approaches (Fig. 7b) considers each label as a separate class, disregarding the hierarchy completely. This approach is among the simplest and resembles a one-versus-rest approach (Koller and Sahami, 1997). The individual classifiers receive hierarchical information from the specific training and testing phases, which we outline in Section 3.1.4. *Local per-level* methods (Fig. 7c) assign a classifier to each relevant category level, which is tasked to decide for that level alone. Finally, *local per-parent* methods (Fig. 7d) assign a classifier to all parent nodes, tasking them to assign a sample to one of its children — therefore capturing part of the sample's current path through the hierarchy.

### 3.1.3. Global classifiers

A global (or *big-bang*) classification approach (Fig. 7e) makes use of the entire hierarchy to make the final classification decision. It is common (though not strictly necessary) for global classifiers to perform actual classification on a flattened representation; hierarchy information is therefore achieved through structural bias (*i.e.*, the architecture of the model) (Silla and Freitas, 2011).



**Figure 7:** The most common approaches to HTC, exemplified on a tree-like hierarchy. Flattened classifiers (7a) lose all hierarchical information, while local classifiers (7b, 7c, 7d) can incorporate some of this information. Global classifiers (7e) aim to fully exploit the label structure.

### 3.1.4. Training and testing local classifiers

It is worth noting that, differently from standard (flat) classification approaches, local classifiers usually require specialized procedures for training and testing, both in terms of actual methods and in the definition of positive and negative examples, as outlined in the following paragraphs.

**Testing** First off, testing phases are usually characterized by a “top-down” flow of information. This is the preferred approach to all local approaches. The system performs a prediction at the first, most generic level, then forwards the decision to the children of the predicted class. As we mentioned, this makes these approaches vulnerable to error propagation (where a mistake at higher levels leads to an inevitable mistake at the lower ones), unless a specialized procedure is set in place to avoid this.

It is worth noting that local classifiers can be used independently, therefore lending themselves naturally to multilabel scenarios. However, this might lead to class-membership inconsistency (by which a children node disagrees with a parent classification); therefore, top-down approaches are usually utilized.

**Training** As local classifiers are defined at different levels of the hierarchy, examples may need to be altered so that the objective makes sense at the local level. Several possible policies may be utilized for the creation of these subsets of examples, each of which differs in how “inclusive” they are. For instance, we might decide whether or not to include samples labeled with “AI” as positive examples for classifiers at the higher “Computer Science” level in Fig. 6a, and consider everything else as a negative example. We point interested readers to Silla and Freitas (2011) for an in-depth overview of these policies.

It is also possible to allow for different classification algorithms in different nodes, an approach which is often attributed to per-parent approaches (Silla and Freitas, 2011). To do this, training data may be further split into sub-train and validation sets, and the best decider for each node is selected dynamically.



### 3.2. Non-mandatory leaf node prediction and blocking

In hierarchical classification datasets, it may not always be the case that all prediction targets correspond to leaves. Many authors distinguish between *Mandatory* and *Non-mandatory Leaf Node Prediction* (MLNP, NMLNP) (Sun and Lim, 2001; Freitas and de Carvalho, 2007). Quite simply, in NMLNP scenarios, the classification method should be able to consider stopping the classification at any level of the hierarchy, regardless of whether it is a leaf node or not. The term applies to both Tree- and DAG-structured hierarchies. In this section, we briefly outline how the different types of HTC approaches can deal with the latter, more complex case.

#### 3.2.1. Flattened classifiers

As mentioned, the flat classification approach simplifies the problem to a standard, non-hierarchical classification problem. In that sense, if the target is restricted to the leaf nodes of the structure, methods that follow this paradigm are unable to deal with NMLNP by design (Silla and Freitas, 2011).

In a naive approach, it is possible to extend the classification targets to include all possible labels, though this would make the task much harder since flattened classifiers do not have any inherent information about the hierarchy. As we will discuss in the following paragraphs, it is possible to inject hierarchical information whilst maintaining the general classification target and algorithm, which is what some global approaches do (Marcuzzo et al., 2022b).

#### 3.2.2. Local classifiers

As we previously discussed, local approaches integrate local information, of which we identified three standard approaches: per-node, per-parent and per-level. To deal with NMLNP, local approaches must implement a *blocking mechanism*, such as to allow to stop inference at any level of the hierarchy. A simple way to do this is by utilizing a threshold on the confidence of each classifier; if during top-down prediction, the confidence doesn't meet the requirement, the inference process is stopped (Ceci and Malerba, 2007).

*The blocking problem* Utilizing thresholds in top-down inference procedures, however, can be problematic, as it may lead to incorrect early stopping of classification. Sun et al. (2004) define this as the *blocking problem*, which refers to any case in which the confidence rating of a parent classifier mistakenly decides an example does not belong to its macro-class. As a consequence, the example will never reach its appropriate sub-class, which will therefore never have a chance to examine it. The authors propose some blocking reduction methods, which generally act by reducing the thresholds of inner classifiers or by allowing lower-level classifiers to have a second look at rejected examples.

#### 3.2.3. Global classifiers

Global classifiers utilize a single, usually complex algorithm that integrates the hierarchy into its internal reasoning. As mentioned, it is common to base global classifiers on an existing flat classification approach and modify it to take into account the class hierarchy. Global classifiers can also be used in NMLNP scenarios, though specific strategies might be needed in this case to enforce class-membership consistency in prediction. Likewise, it is also possible to integrate a top-down prediction approach (which would be internal to the algorithm) to avoid this issue. Overall, since global classifiers are customarily built to address certain hierarchical structures, they can be specialized to allow for NMLNP tasks.

### 3.3. Evaluation measures

As HTC issues are inherently multiclass (or multilabel), many researchers choose to utilize standard evaluation metrics widely adopted in classification scenarios. As mentioned, however, many authors argue that these measures are inappropriate (Sun and Lim, 2001; Kiritchenko et al., 2006; Kosmopoulos et al., 2015; Stein et al., 2019; Silla and Freitas, 2011). Intuitively, these arguments are based on the shared belief that ignoring the hierarchical structure in the evaluation of a model is wrong because of the concept of *mistake severity*; in other words, a model that performs “better mistakes” should be preferred (Vaswani et al., 2022). This follows from two considerations. Firstly, predicting a label that is structurally close to the ground truth should be less penalizing than predicting a distant one. Secondly, errors in the upper levels of the hierarchy are inherently worse (e.g., misclassifying “football” as “rugby” is comparatively better than misclassifying “sport” as “food”).

In this section, we will provide an outline of the most common evaluation metrics utilized in HTC, both standard and hierarchical. For a more in-depth analysis of these metrics, as well as of the issues they present and how to address them, we refer to Kosmopoulos et al. (2015).

### 3.3.1. Standard metrics

The most common performance measures utilized in “flat” classification are derived from the classic information retrieval notions of *Accuracy* ( $Acc$ ), *Precision* ( $Pr$ ), and *Recall* ( $Re$ ). As in any other supervised learning task, we consider the truthfulness of a model’s predictions against the *ground truth* derived from the dataset. Formally, let  $\{(x_0, y_0), \dots, (x_n, y_n)\}$  be a set of labeled training examples, where  $x \in \mathbb{R}$  is an input example and  $y \in \{0, 1\}^L$  is the associated label vector, with  $L$  being the set of label indices (therefore,  $|L|$  is the number of categories). For each label  $l \in L$ , we can calculate category-specific metrics by considering positive (P) and negative (N) predictions for each example. A prediction is considered true (T) if it agrees with the ground truth, and false (F) otherwise.

*Accuracy* measures the ratio of correct predictions over the total of number predictions. *Precision* is instead a measure of *correctness*, quantifying the proportion of true positive predictions among the ones made, while *Recall* is a measure of *completeness*, quantifying the proportion of overall positives captured by the model. Notably, the latter two metrics have a larger focus on the impact of false predictions. Accuracy, Precision, and Recall are defined as follows (Eq. 14):

$$Acc = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \quad Pr = \frac{|TP|}{|TP| + |FP|} \quad Re = \frac{|TP|}{|TP| + |FN|} \quad (14)$$

It is worth mentioning that Precision and Recall do not effectively measure classification performance in isolation (Sebastiani, 2002). Therefore, a combination of the two is commonly utilized. The F-measure ( $F_\beta$ ) is the most popular of these combinations, providing a single score according to some user-defined importance of Precision and Recall (*i.e.*,  $\beta$ ). Normally,  $\beta = 1$ , resulting in the harmonic mean of the two measures (Eq. 15):

$$F_\beta = \frac{(\beta^2 + 1) \cdot Pr \cdot Re}{\beta^2 \cdot Pr + Re} \quad F_1 = 2 \cdot \frac{Pr \cdot Re}{Pr + Re} \quad (15)$$

While Accuracy naturally extends to multiclass settings without being able to weigh the contribution of class differently, Precision and Recall might be averaged in different ways. In particular, *macro* averaging considers all class contributions equally (Eq. 16), while *micro* averaging treats all examples equally (Eq. 17):

$$Pr_{macro} = \frac{\sum_{i=1}^m Pr_i}{m} \quad Re_{macro} = \frac{\sum_{i=1}^m Re_i}{m} \quad (16)$$

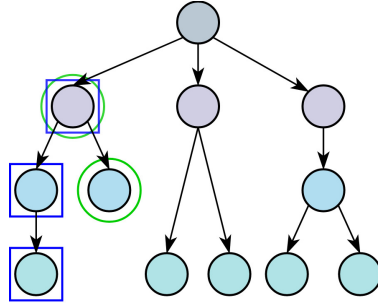
$$Pr_{micro} = \frac{\sum_{i=1}^m |TP_i|}{\sum_{i=1}^m |TP_i| + |FP_i|} \quad Re_{micro} = \frac{\sum_{i=1}^m |TP_i|}{\sum_{i=1}^m |TP_i| + |FN_i|} \quad (17)$$

Where  $m$  is the number of categories. Micro averaging may be useful when class imbalance is severe and needs to be accounted for in the measurements. Support-weighted metrics are also an alternative in such cases. Lastly, F-measures may be micro or macro-averaged as well, utilizing the corresponding averaged versions of Precision and Recall.

### 3.3.2. Hierarchical metrics

As outlined before, standard metrics lack the capability of reflecting the relationships that exist among classes. In this context, Sun and Lim (2001) propose to solve this issue by introducing hierarchical metrics based on category *similarity* and *distance*. Category similarity evaluates the cosine distance of the feature vectors representing predicted and true categories, while category distance considers the number of links between the two in the hierarchy structure. While interesting, these measures have practical issues, which Kiritchenko et al. (2006) outline (such as inapplicability to DAGs and multilabel tasks). Instead, they propose to extend traditional precision and recall instead. To do this, they augment the set of predicted and true labels to include all their ancestors (Fig. 8) and calculate the metrics on the augmented sets.

Formally, let  $\hat{Y}_{aug}$  be the augmented set of predictions, containing the most specific predicted classes and all of their ancestors, and  $Y_{aug}$  the augmented set of the most specific ground truth class(es) and all of their ancestors. Then, hierarchical metrics (h-metrics) can be defined as follows (Eq. 18):



**Figure 8:** Tree hierarchy with predicted (squares) and ground truth (circles) labels, with the ancestors of each highlighted. As the two sets have a node in common, the misprediction should be considered less severe.

$$hPr = \frac{\sum_i |\hat{Y}_{aug} \cap Y_{aug}|}{\sum_i |\hat{Y}_{aug}|} \quad hRe = \frac{\sum_i |\hat{Y}_{aug} \cap Y_{aug}|}{\sum_i |Y_{aug}|} \quad hF_\beta = \frac{(\beta^2 + 1) \cdot hPr \cdot hRe}{\beta^2 \cdot hPr + hRe} \quad (18)$$

However, Kosmopoulos et al. (2015) observe that this approach overpenalizes errors in which nodes have many ancestors. Specifically, false positives predicted at lower levels of the hierarchy strongly decrease the precision metric, while recall tends to be boosted, since adding ancestors is likely to increase the number of true positives (false positives are ignored). Drawing from the concept of lowest common ancestor (LCA) as defined in graph theory (Aho et al., 1976), they propose LCA-based measures as a solution. Briefly, and in the context of tree structures, the LCA of two nodes can be defined as the lowest (furthest from the root) node that is an ancestor of both. Therefore, these new measures are defined similarly to the one in Eq. 18, but with the expanded sets defined in terms of LCA rather than on full node ancestries (Stein et al., 2019). Despite these issues, the hierarchical metrics of Eq. 18 are still considered effective measures across a broad range of scenarios (Silla and Freitas, 2011).

Some authors report metrics that explicitly evaluate the difference, in terms of path distances through the hierarchy, between class predictions and ground truth labels. In particular, Sainte Fare Garnot and Landrieu (2021) define the Average Hierarchical Cost (AHC) of a set of predictions. Given two labels  $a, b \in L$ , define  $\text{dist}(a, b)$  as the number of edges that separate two labels in the hierarchy tree. In turn, for a set of labels  $S$ , let  $\text{dist}(a, S)$  be the minimum distance between the node and the set, *i.e.*,  $\min(\text{dist}(a, s) \forall s \in S)$ . If predictions and ground truth labels are expressed as vectors  $\in \{0, 1\}^L$  (as before), the AHC can be defined as (Eq. 19):

$$AHC(Y, \hat{Y}) = \frac{1}{|Y|} \sum_j \hat{y}_j \text{dist}(j, Y) \quad (19)$$

In practice, this measure evaluates how “far off” the predictions were from the closest common ancestor (and, in that, are not too dissimilar for LCA-based measures). As a consequence, a low AHC indicates that mistakes were, on average, not too distant from the ground truth.

### 3.3.3. Other metrics

Depending on the method and the particular aspect of HMC being tackled, authors may utilize different metrics in evaluating hierarchical methods. In this section, we briefly introduce them.

Some authors report rank-based evaluation metrics, which are widely used in the more general context of multilabel classification (Chen et al., 2020; Gong et al., 2020). Among many, the two most commonly utilized are Precision and Normalized Discounted Cumulative Gain at  $k$  ( $Pr@k$ ,  $NDCG@k$ ). Given a model’s prediction (*i.e.*, a probability vector across labels), we can sort the labels in descending based on the output probabilities. Then,  $Pr@k$  indicates the fraction of correct predictions in the top- $k$  labels of this sorted list, whereas  $NDCG@k$  measures ranking quality (*i.e.*, how well the list is sorted, therefore how high are correct labels ranked). For a more precise description of these metrics, which are often utilized in recommendation systems and, more in general, in information retrieval, see Marcuzzo et al. (2022a).

In methods able to categorize labels at different levels of the hierarchy separately, many authors chose to showcase the accuracy score at each separate level, as well as a single overall score (Ma et al., 2020; Wang et al., 2021b). The overall score is the one obtained by classifying the last level of the hierarchy given the (possibly incorrect) predictions of the parent classes. In some cases, authors utilize *subset* accuracy, where all labels must match the ground truth exactly. Lastly, some authors utilize the *Hamming loss* (Dong et al., 2022; Yu et al., 2021) metric. This measure evaluates the fraction of misclassified instances, *i.e.*, true labels that were not predicted or predictions that were not in the ground truth. A comprehensive review of these metrics, as well as less commonly utilized ones, may be found in Zhang and Zhou (2014).

## 4. Hierarchical Text Classification Methods

In this section, we provide an overview of approaches devised to solve HTC tasks in the 2019–2022 period (for reasons previously discussed). Tables include all relevant works we have found in our literature review, while we will analyze in more detail a subset of these proposals in the main body of the section.

### 4.1. Recent proposals

Tables 1, 2, and 3 display the results of our search for the latest applications of HTC to the NLP domain. While we do not discuss them, it should be mentioned that there are also other domains of application of HMC that have much in common with HTC, and one could also draw inspiration from such works (for example, protein function prediction in functional genomics) (da Silva and Cerri, 2021; Stepišnik and Kocev, 2020; Cerri et al., 2019; Romero et al., 2022).

### 4.2. Analyzed methods

In this section, we go into further detail in describing a subset of the recent proposals summarized by Tables 1, 2, and 3. As the number of works is large, we limit ourselves to methods that report results on the two most common datasets, which are listed in Table 7. As we will discuss in Section 5, these are also the methods we considered when gathering implementations to test for the experimental part of this survey.

*HTrans* In their work, Banerjee et al. (2019) propose Hierarchical Transfer Learning (HTrans), a framework to improve the performance of a local per-node classification approach. The general intuition is that knowledge may be passed to lower-level classifiers by initializing them with the parameters of their parent classifiers. First, they utilize a bidirectional GRU-based RNN enhanced with an attention mechanism as a text encoder, and then use a fully connected network as a decoder to produce the class probability. Word embeddings are initialized with GloVe pre-trained embeddings. One such model is trained for each node in the hierarchy tree with binary output, and child nodes shared parameters with the classifiers of ancestor nodes. This can be seen as a “hard” sharing approach, which utilizes fine-tuning to enforce inductive bias from parent to child nodes. The inference is achieved through a standard top-down approach. The authors perform an ablation study by removing parameter sharing and attention, and also compare results with a multilabel model initialized with weights from the binary classifiers, showcasing solid improvements when including their proposed enhancements.

*HiLAP* Mao et al. (2019) tackle the issues that arise from a mismatch between training and inference in local HTC approaches. They propose a reinforcement-learning approach as a solution, modeling the HTC task as a Markov Decision Process; the task consists in learning a policy that considers where to place an object (which label) as well as when to stop the assignment process (allowing for NMLNP). In other words, such a policy allows the algorithm to “move” between labels or “stop” when necessary. Though theoretically extendable to any neural encoder, the authors use TextCNN (Kim, 2014) with GloVe vectors and BoW features in the document encoder to produce fixed-size embeddings. They also create randomly initialized embeddings for each label, producing an “action embedding” matrix, as each label defines an action for the agent. The encoded document is concatenated with the embedding of the currently assigned label to produce the current state vector. After passing it through a two-layer fully connected network with ReLU activation, the state vector is multiplied by the action embedding matrix to determine the probabilities of all possible actions. In the beginning, each document is placed on the root node, and the assignment stops when the “stop” action is selected as the most probable. The loss function is defined in terms of the overall  $F_1$  score between the previous and the current time step. The proposed Hierarchical Label Assignment Policy (HiLAP) yields excellent results, in particular in terms of consistency of parent-child assignments with respect to the hierarchy of classes.

**Table 1**

Hierarchical Text Classification methods. Names for models are reported when authors provide them.

Method	Base	Datasets	Code	Novelty
HCCMN (Liu et al., 2019a)	TCN, LSTM, ATT	CRTEXT, SogouCA, Fudan, TouTiao	✗	Combine TCN with LSTM for extraction of contextual information and temporal features from Chinese text
(Yang and Liu, 2019)	CNN, LSTM, ATT	RCV1, AAPD, Zhihu-QT	✗	Usage of local and global information, new Seq2seq model with CNN encoder and LSTM decoder
HLSE (Gargiulo et al., 2019)	CNN	MeSH	✗	Hierarchical Label Set Expansion for labels regularization
(Masoudian et al., 2019)	k-means, SVM	Hamshahri	✗	HTC of Persian text with weak supervision using clustering
(Li et al., 2019)	SVM, MLP	-	✗	Mixed deep-learning and traditional methods for HTC in the automotive domain
WeSHClass (Meng et al., 2019)	CNN	NYT, AAPD, Yelp	✓	Weakly supervised HTC with pseudo-document generation
(Aly et al., 2019)	Caps	BGC, WOS	✓	Compare CapsNets with other NNs
HTrans (Banerjee et al., 2019)	GRU, ATT	RCV1	✗	Recursive training of LMs on branches of tree-shaped hierarchies of labels
NeuralClassifier (Liu et al., 2019b)	-	RCV1, Yelp	✓	Toolkit to quickly setup HTC pipelines with multiple algorithms
HARNN (Huang et al., 2019)	RNN, ATT	Patent	✓	Hierarchical Attention Unit (HAM)
HiLAP (Mao et al., 2019)	BASE	RCV1, NYT, Yelp	✓	RL framework to learn a label assignment policy
HLAN (Caled et al., 2019)	LSTM, ATT	EURLEX, EURLEX-PT	✓	HTC of legal documents, new Portuguese corpus
(Prabowo et al., 2019)	RF, DT, NB, SVM	HS-Ind	✓	Categorization of hate speech expression in Indonesian tweets
HKUST (Xiao et al., 2019)	BASE	RCV1, 20NG	✓	Path prediction method
NETHIC (Ciapetti et al., 2019; Lomasto et al., 2020)	MLP	DANTE	✓	DNN for efficient and scalable HTC
LSNN (Xu and Du, 2020)	MLP	20NG, DBpedia, YahooA	✗	Unsupervised clustering to exploit labels relation, neural architectures to learn inter- and intra- label relations
HG-Transformer (Gong et al., 2020)	TRAN	RCV1, Amazon	✗	Transformer-based model, weighted loss using semantic distances
H-QBC (Nakano et al., 2020)	QBC (AL)	Enron, Reuters	✗	Active learning framework for HTC, for classification of dataset with few labeled samples
GMLC (Zhao et al., 2020)	LSTM, ATT	-	✗	Seq2seq multi-task framework with hierarchical mask matrix to learn cross-task relations
3LBT (Addi et al., 2020)	SVM, NB, DT, RF	HARD	✗	Framework for sentiment classification of Arabic texts using multi-level hierarchical classifiers, using SMOTE technique
(Jiang et al., 2020)		FIN-NEWS-CN	✗	Weakly supervised model driven by user-generated keywords, adopting a confidence-enhancing method during training
C-HMCNN (Giunchiglia and Lukasiewicz, 2020)	MLP	Enron, 19+	✓	Coherent predictions using a hierarchical constraint, usage of hierarchical information

**MATCH** The authors of MATCH (Zhang et al., 2021) propose to boost the multilabel classification performance by learning a text representation enriched with document metadata and further adding a regularization objective to exploit the label hierarchy. The first component of their architecture is a metadata-aware pre-training scheme, that jointly learns words and metadata embeddings considering the vicinity between documents and related metadata and labels. A modified Transformer encoder is then used to compute document representations: in order to cope with large label spaces, several special tokens ([CLS]) are pre-pended to input sequences, followed by the metadata tokens and the document's words. The representation of all [CLS] tokens is then concatenated and passed into a fully connected layer with a sigmoid activation function. An L2 regularization (hypernymy regularization) is applied to the classification layer (*i.e.*, to the weight matrix), forcing the parameters of each label to be similar to the ones of its parent. For all pairs



**Table 2**

Hierarchical Text Classification methods. Names for models are reported when authors provide them.

Article	Base	Datasets	Code	Novelty
HyperIM (Chen et al., 2020)	GRU	RCV1, Zhihu, Wik-iLSHTC	✓	Embedding of label hierarchy and document in the same hyperbolic space, with explicit modeling of label-word semantics
LCN (Krendzelak and Jakab, 2020)	CNN	20NG	✗	Different approaches with CNNs for HTC
(Rojas et al., 2020)	GRU, ATT	WOS, DBpedia	✗	Seq2seq model, auxiliary tasks, and beam search
HiAGM (Zhou et al., 2020b)	LSTM, GCN	RCV1, WOS, NYT	✓	Extraction of hierarchical label information
ONLSTM (Ma et al., 2020)	LSTM	WOS, DBpedia	✓	Sharing parameters for sequential prediction of next-level label
F-HMTC (Liang et al., 2020)	BERT	-	✓	BERT embeddings with hierarchy-based loss
(Li et al., 2020)	TRAN	B-SHARP	✗	Hierarchical Transformer for classification based on ensemble of three models
CorNet (Xun et al., 2020)	BERT, CNN, RNN	EURLEX, Amazon, Wikipedia	✓	Model-agnostic architecture for MLTC to enhance label predictions to consider the labels' correlation
MAGNET (Pal et al., 2020)	LSTM, GAT	Reuters, RCV1, AAPD, Slashdot, Toxic	✓	Improving GCN with GAT for considering label correlation and co-occurrence
PAC-HCNN (Zhu et al., 2020)	CNN, ATT, GLU	-	✗	Stacked hierarchical-convolutional layers for HTC of Chinese patent dataset
(Masmoudi et al., 2021)	Co-training	-	✗	Semi-supervised approach for HTC of research papers, based on Co-training algorithm
JMAN (Dong et al., 2021)	GRU	Bibsonomy, Zhihu, CiteULike	✓	Attention mechanism to mimic user annotation behavior, loss regularization to capture label co-occurrence relations
MATCH (Zhang et al., 2021)	TRAN (CorNet)	MAG-CS, PubMed	✓	E2E framework for classification with large hierarchies of labels and documents' metadata
(Ye et al., 2021)	TRAN, GNN	MAG-CS, PubMed	✗	Incorporate text metadata and labels structure using heterogeneous graphs
RLHR (Liu et al., 2021)	BERT, RL	WOS, Yelp, QCD	✓	Label structure modeling using Markov decision process and RL with pre-trained LMs for zero-shot TC
HCSM (Wang et al., 2021a)	Caps, LSTM	RCV1, EURLEX, WOS	✗	Combine global label structure and partial knowledge learning to emulate cognitive process
HiMatch (Chen et al., 2021)	BERT, GCN, GRU, CNN	RCV1, EURLEX, WOS	✓	Semantic matching method to learn relationship between labels and text
HIDDEN (Chatterjee et al., 2021)	CNN	RCV1, NYT, Yelp	✓	Embed hierarchical label structure in label representation jointly training the classifier
HE-AGRCNN (Peng et al., 2021)	CNN, ATT, Caps, RNN	RCV1, 20NG	✓	Architecture, graph-of-words text representation, taxonomy-aware margin loss
CoPHE (Falis et al., 2021)	n/a	MIMIC-III	✓	New metrics for hierarchical classification for large label spaces
CLED (Wang et al., 2021b)	Caps, CNN, GRU	DBpedia, WOS	✓	Concept-based label embeddings to model sharing in hierarchical classes
HMATC (Aljedani et al., 2021)	Tree-based (HOMER)	MLAD	✓	HTC in multi-label setting for Arabic text, introduction of new Arabic corpus
ICD-RE (Tsai et al., 2021)	AE, TRAN	MIMIC	✓	Re-ranking method to learn label co-occurrence with unknown label hierarchies

of parent ( $l$ ) and child labels ( $l'$ ) the regularization term is expressed as:

$$R_{\text{parameter}} = \sum_{l \in \mathcal{L}} \sum_{l' \in \sigma(l)} \frac{1}{2} \|\mathbf{w}_l - \mathbf{w}_{l'}\|^2 \quad (20)$$

Where  $\mathcal{L}$  is the set of labels,  $\sigma(l)$  is the set of parent labels of  $l$ , and  $\mathbf{w}_l$  denotes the parameters of a label  $l$ . The final predictions are also regularized to penalize the model when the probability of a parent label is smaller than the one for

**Table 3**

Hierarchical Text Classification methods. Names for models are reported when authors provide them.

Article	Base	Datasets	Code	Novelty
SASF (Zhao et al., 2021)	GCN, GRU, CNN	WOS, BGC	✗	Hierarchy-aware label encoder and attention mechanism applied to document labels and words
L1-L2-3BERT (Yu et al., 2021)	BERT	-	✗	Hierarchical classification using multi-task training with penalty loss
HierSeedBTM (Yang et al., 2021)	HuffPost, 20NG	-	✗	Dataless hierarchical classification with iterative propagation mechanism to exploit hierarchical labels' structure
HiDEC (Im et al., 2021)	ATT	WOS, RCV1	✗	Recursive hierarchy encoder-decoder with self-attention to exploit dependency between labels
THMM (Pujari et al., 2021)	ATT	Patent	✗	Pre-trained LM with multi-task architecture for efficient HTC
TaxoClass (Shen et al., 2021)	RoBERTa	Amazon-531, DBPedia-298	✗	HMTTC with class surface name as the only supervision signal
HTCInfoMax (Deng et al., 2021)	HiAGM-LA	RCV1, WOS	✓	Text-label mutual information maximization and label prior matching
PAMM-HiA-T5 (Huang et al., 2021)	T5	RCV1, NYTimes, WOS	✗	Hierarchy aware T5, path-adaptive mask mechanism
(Ma et al., 2020)	LSTM	WOS, DBpedia	✗	Hierarchical fine-tuning approach with joint embedding learning based on labels and documents
HPT (Wang et al., 2022b)	LSTM	RCV1, WOS, NYT	✗	Hierarchy-aware masked language modeling, suitable to handle labels imbalance
HE-HMTC (Ma et al., 2022)	BiGRU, SDNE	WOS, Amazon, DBpedia, WebService, BestBuy	✓	Level-by-level hybrid embeddings, graph embedding of category structure
LA-HCN (Zhang et al., 2022)	ATT	WIPO-alpha, BGC, Enron, RCV1	✗	Label-based attention, local and global text embedding
HGCLR (Wang et al., 2022a)	TRAN	RCV1, WOS, NYT	✓	Contrastive learning, Graph-based Transformers for sample generation
Seq2Tree (Yu et al., 2022)	T5	RCV1, WOS, BGC	✗	Decoding strategy to ensure consistency in predicted labels across levels, new metrics for HTC
HBGL (Jiang et al., 2022)	BERT	RCV1, WOS, NYT	✓	Hierarchical label embeddings encoding labels structure using both local and global hierarchical data
(Song et al., 2022)	MF, ATT	-	✗	Recursive attention mechanism to capture relations between labels at different levels, applied to the MF algorithm
CHAMP (Vaswani et al., 2022)	n/a	RCV1, NYT	✗	Definition of new metrics and modified BCE loss for hierarchical classification
OWGC-HMC (Dong et al., 2022)	MLP	Aizhan	✗	Online classifier for genre classification based on entity extraction and embedding with hierarchical regularization term
HMC-CLC (Xu et al., 2022)	BN	RCV1, Enron, imclef07a	✗	Exploit labels correlations for HTC with greedy label selection method to determine correlated labels dynamically
ML-BERT (Marcuzzo et al., 2022b)	BERT	Linux Bugs	✓	Global approach with multiple BERT models trained on different hierarchy levels, test of document embedding strategies

the child label. The value is summed over all documents  $d \in \mathcal{D}$ , as in the equation below:

$$R_{\text{output}} = \sum_{d \in \mathcal{D}} \sum_{l \in \mathcal{L}} \sum_{l' \in \sigma(l)} \max(0, \pi_{dl} - \pi_{dl'}) \quad (21)$$

Where  $\pi_{dl}$  represents the probability that document  $d$  belongs to child class  $l$ . The authors report the results of an ablation study that confirms the metadata embedding strategy and hierarchy-aware regularization are both beneficial to the classification task.

*HiAGM* Zhou et al. (2020b) propose an end-to-end hierarchy-aware global model (HiAGM) that leverages fine-grained hierarchy information and aggregates label-wise text features. Intuitively, they aim to add information to traditional text encoders by introducing a hierarchy-aware structure encoder (the structure being the hierarchy). As structure encoders, the authors test a TreeLSTM and a GCN adapted to hierarchical structures. Moreover, they propose

two different frameworks: one based on multi-label attention (HiAGM-LA), and one on text feature propagation (HiAGM-TP). HiAGM-LA utilizes the attention mechanism to enhance label representations in a bidirectional, hierarchical fashion, utilizing node outputs as the hierarchy-aware label representation. HiAGM-TP, on the other hand, is based on text feature propagation in a serial dataflow; text features are used as direct inputs to the structure encoder, propagating information throughout the hierarchy. For multilabel classification the Binary Cross Entropy (BCE) loss is used, as well as the regularization term  $R_{\text{parameter}}$  that was described for MATCH.

**RLHR** The approach used by Liu et al. (2021), which tackles zero-shot HTC, includes a reinforcement learning (RL) agent that is trained to generate deductions path, *i.e.*, the possible paths from the root label to a child label, to introduce Hierarchy Reasoning (HR). The reward is assigned depending on the correctness of the predicted paths, which should be sub-paths of the ground truth set of paths  $t$  positively reward the agent. Moreover, the authors design a rollback algorithm that overcomes the inefficiencies of previous solutions and allows the model to correct inconsistencies in the predicted set of labels at inference time. The zero-shot task is formulated as a deterministic Markov Decision Process over label hierarchies. BERT and DistilBERT are used as base models, which are pre-trained on a binary classification task with a negative sampling strategy. Several training examples are created by pairing each document with one of its labels as well as a number of irrelevant labels to provide positive and negative examples. Using the pre-trained model, a policy is learned to further tune the model on the binary classification task.

**HiMatch** In HiMatch (Chen et al., 2021), the HTC task is framed as a semantic matching problem, and the method is used to jointly learn embeddings for documents and labels and to learn a metric for matching documents with labels. The proposed architecture first utilizes a text encoder akin to the one used by HiAGM. Then, a *label encoder* is used to produce label embeddings enriched with dependencies among labels. It uses the same GCN architecture as the text encoder, and label vectors are initialized with BERT's pre-trained embeddings. The document and label representations are used in the label semantic matching component, which projects text and labels into a common feature space using two independent two-layer feed-forward networks. A cross-entropy objective is used for training with two regularization terms: one to force documents and respective labels to share a similar representation, measured in terms of mean square error, and a second one to penalize close semantic distance between a document and incorrect labels. The latter constraint uses a triplet margin loss objective with euclidean distance. All components are trained jointly, and the authors report improved performance over a BERT-based model fine-tuned on multilabel classification.

**HE-AGCRCNN** The usage of CNs has also been proposed for HTC tasks. Aly et al. (2019), for instance, adapt a CN to exploit the labels' co-occurrence, correcting final predictions to include all ancestors of a predicted label such as to ensure consistent predictions. More recently, Peng et al. (2021) discuss the drawbacks and strengths of several popular neural networks used for text processing, including CNN, RNN, GCN, and CNs. They propose to combine them in a single architecture (AGCRCNN) so that they can better capture long- and short-range dependencies and both sequential and non-consecutive semantics. In their work, documents are modeled using a graph of words that retains word ordering information: after a lemmatization and stop-word removal step, each word is represented as a node with their position in the document set as an attribute. A sliding window is passed over the document and edges between nodes are created, weighted on the number of times a word co-occurred in all sliding windows. For each document, they extract a sub-graph whose nodes are the document's words with the highest *closeness centrality* — a measure of the importance of words in a document — and their neighbor nodes, up to a fixed number. Nodes are then mapped to Word2Vec embeddings, obtaining a 3D representation that is fed to the Attentional Capsule Recurrent CNN module. This is composed of two blocks, each one containing a convolutional layer to learn higher-level semantic features, and a LSTM layer with attention mechanism to learn local sequentiality features specific to each sub-graph. A CN with dynamic routing mechanism is finally used for multilabel classification. They perform an ablation study comparing several variations of their architecture, as well as previously proposed deep learning classifiers, showcasing better results for their proposed model on two datasets.

**CLED** As another example of the application of CNs to HTC, Wang et al. (2021b) propose the Concept-based Label Embedding via Dynamic routing (CLED), in which a CN is used to extract concepts from text documents. Concepts can be shared between parent and child classes, and can thus be used to support classification based on hierarchical relations. The top- $n$  keywords from each document are used as concepts and encoded with GloVe word embeddings. A clustering procedure is utilized to initialize concepts' embeddings with the clusters' center. Dynamic routing is

then used to learn concepts' embeddings; agreement is only measured between capsules representing parent and child classes.

**ICD-Reranking** Tsai et al. (2021) tackle the task of automatic ICD coding in medical settings, which requires multilabel classification of clinical nodes in hierarchically dependent diagnostic codes. The authors propose to pair a base predictor (responsible for the generation of top- $k$  most probable label sets) with a re-ranking step. In particular, it is the re-ranker that is designed to capture correlation and co-occurrence between labels. They propose two agnostic re-ranking methods, which they validate across different base predictors to prove the generalizability of their proposed re-rankers. Both re-ranking methods are trained on the same training data as the candidate generator. The first approach, MADE, uses the joint probability as a way to score label sets, estimating it by decomposing it in an autoregressive fashion with random ordering. As this approach may fail to capture non-sequential dependencies, they also propose Mask-SA, a self-attentive approach inspired by NLP's masked language models which estimates a distribution over the label vocabulary for the masked input given all other elements in the set. The authors showcase a consistent improvement across three different predictors on two datasets.

**HGCLR** Wang et al. (2022a) argue against the encoding of text and label hierarchy separately, and instead, propose to aggregate the two representations. They emphasize the fact that the label hierarchy is static, thus translating to an individual representation by the graph encoder. As the interaction becomes redundant, they argue for the direct injection of this representation into the text encoder. The hierarchy-aware representation is achieved through a contrastive learning approach focused on generating positive examples that are both label-guided and hierarchy-involved. The construction of such examples is driven by the observation that a select number of keywords are sufficient to attach a label; similar to an adversarial attack, then, new examples might be created by modifying tokens within the text. However, the aim is not to disrupt the example like in an adversarial attack, but rather to modify unimportant tokens to keep the classification results unchanged. From a technical perspective, important keywords are defined by evaluating the attention weight of token embeddings, while graph embeddings are obtained through a modified Graphormer (Ying et al., 2021).

**HIDDEN** Chatterjee et al. (2021) devise a way to create label embeddings by leveraging the properties of hyperbolic geometry, which has been found to be helpful in the representation of organized structures (like hierarchies). They adopt a specific hyperbolic model, the Poincaré ball model. Briefly, in this model, the distance between pairs of points falls exponentially as one moves from the origin toward the surface of the ball. The authors claim that this property can be used to represent arbitrarily large hierarchies, where the root of the hierarchy can be thought of as being close to the origin, and the leaves lie on the surface of the ball. A TextCNN (Kim, 2014) is used to learn document embeddings, while the hyperbolic model is used to learn label embeddings through a document-label alignment criterion based on the dot product between embeddings. Additionally, a second loss term is used to push labels closer in the hyperbolic embedding space based on their co-occurrence. Their results suggest that the joint learning objective increases performance metrics as compared to a model sequentially optimized on the two objectives.

**CHAMP** Vaswani et al. (2022) propose a modified BCE loss definition that accounts for the severity of mispredictions, which they name Comprehensive Hierarchy Aware Multi-label Predictions (CHAMP). More specifically, the CHAMP loss function penalizes false positives depending on the distance between the incorrectly predicted label and the true labels. On the other hand, false negatives are always considered equally severe. For a set of labels  $L$ , a ground truth vector  $y \in \{0, 1\}^L$ , and a prediction  $\hat{y} \in \{0, 1\}^L$ , the loss function is defined as:

$$\text{CHAMP}(y, \hat{y}) = - \sum_{j=1}^{|L|} y_j \log \hat{y}_j + (1 + s_S(j)) (1 - y_j) \log(1 - \hat{y}_j) \quad (22)$$

where  $s_S(j)$  is the measure of severity of false positive label  $j$ , which depends on the distance from the ground truth label set  $S$ . The severity, where  $\text{dist}(j, S)$  is the minimum distance between  $j$  and any label in  $S$ , can be defined as:

$$s_S(j) = \beta \cdot \frac{\text{dist}(j, S)}{\text{dist}_{max}} \quad (23)$$

Being an alternative loss function, this approach is natively model-agnostic. The authors found substantial improvements in terms of area under Precision-Recall (PR) curve (AUPRC) and retrieval metrics over their baselines trained using the BCE loss on several datasets.

*HE-HMTC* Ma et al. (2022) propose HE-HMTC, a local per-level approach where a different model is used at each level of the hierarchy of labels. First, a bidirectional GRU encoder is used for text representation. Forward and backward hidden states are concatenated to obtain an encoded document. A Structural Deep Network Embedding (Wang et al., 2016) graph embedding method is used to obtain compressed representations for each label in the hierarchy. Specifically, SDNE is an auto-encoder trained separately to reconstruct the adjacency matrix of the DAG that is used to represent the hierarchy and allows to capture both structural and semantic features of the label set. At each level, the classifier receives the text representation from the encoder at the previous level, as well as the previous label embedding. This vector is finally passed through a fully-connected layer and a softmax activation, generating the final probabilities for a level of classes. The process stops when the last level of labels is reached. They validate the model on several datasets and perform an ablation study to confirm the impact of the graph encoder and different text encoding schemes.

*GCN-HT* Ferdowsi et al. (2021) propose GCN-HT, a GCN-based approach with message passing applied to the classification of free text into stages of clinical trial (CT) protocols. Textual data is encoded to a feature vector using both BoW and BERT embeddings. CT codes are represented in a graph and a GCN is used to update their representation. In particular, they show that their proposed selective graph pooling operation can lead to performance improvements, as parts of the hierarchy are invariable across different documents.

*HTCInfoMax* Deng et al. (2021) build upon the work done by Zhou et al. (2020b), adding two modules on top of the HiAGM model. They use the probability distribution produced by the encoder and use a discriminator to estimate the mutual information between a document and its labels. This module is used to clean documents of irrelevant label information and to embed documents with corresponding labels' information. The second module is used to constrain the label encoder to learn better label representations, especially for low-frequency labels. They use a loss function to maximize the mutual information between text and labels, as well as a regularization term that pushes the learned distribution of labels close to its true distribution. The authors showcase superior performance to HiAGM on two datasets, as well as demonstrating the effectiveness of each module through an ablation study.

### 4.3. Datasets used in HTC literature

We conclude this section on HTC literature by providing an overview of the most commonly utilized datasets in the context of HTC, as inferred by analyzing recent methods in the previous sections. In Table 4 we summarize datasets that we both encountered frequently and are also well defined. Conversely, in Table 5 we list large collections from which HTC datasets are often derived, but without there being a consistent set of data utilized each time.

Indeed, we found much of HTC literature is spread across a wide variety of datasets, which often collect data from the same source but utilize it in different ways. While what we report in Table 5 are among the most prominent sources in terms of raw data, there are others, such as with data derived from DBpedia ([dbpedia.org](http://dbpedia.org) [Internet], accessed 2022 Nov 15) and Wikipedia ([wikimedia.org](http://wikimedia.org) [Internet], accessed 2022 Nov 15) dumps. Overall, methods listed in Tables 1, 2, and 3 span across 44 different datasets, many of which are only tested on the specific method being proposed, at least recently. Therefore, HTC suffers from a lack of established benchmarks, resulting in this scattering of methods over a wide range of incomparable datasets.

Furthermore, some of these datasets provide pre-defined splits, such as the very popular RCV1 dataset (which is by far the most utilized dataset, followed by the WOS dataset). However, when comparing results with methods, one should be mindful to check whether the authors have made use of such splits; indeed, the most common split of the RCV1 dataset utilizes a very small portion of overall data for training (around 3%). With this in mind, methods that adopt larger training splits (which we found in some works) should be compared to others with due precaution.

## 5. Experiments and Analysis

This section presents the experimental part of this work: first, we describe our selection of benchmark datasets, then proceed to introduce the baseline methods we tested and finally we discuss and compare their performance.



**Table 4**

Commonly utilized datasets.

Name	Size	Depth	Labels (overall)	Labels per level
RCV1-v2 (Lewis et al., 2004)	804,414	4	103	4-55-43-1
Web of Science (WOS-46,985) (Kowsari et al., 2018)	46,985	2	145	7-138
Blurb Genre Collection (BGC) (Aly et al., 2019)	91,894	4	146	7-46-77-16
20Newsgroup (20NG) (Lang, 1995)	18,846	2	20	6-20
Arxiv Academic Paper (AAPD) (Yang et al., 2018; Xu et al., 2021)	55,840	2	61	9-52
Enron (Klimt and Yang, 2004; Zhang et al., 2022)	1,648	3	56	3-40-13
Patent / USPTO (Huang et al., 2019)	100,000	4	9,162	9-128-661-8,364

**Table 5**

Large collections from which HTC datasets are often derived.

Name	Size	Depth (overall)	Labels (overall)
New York Times (NYT) (Sandhaus, 2008)	~ 1,8M	10	2,318
YELP (Mao et al., 2019; Chatterjee et al., 2021)	~ 7M	2	539
Amazon (McAuley and Leskovec, 2013)	~ 35M	3	531

**Table 6**

Statistics for datasets used in this work.

	Bugs	RCV1-v2	WOS	BGC	Amazon
Size	35,050	804,414	46,960	91,894	500,000
Depth	2	4**	2	4	2
Labels overall	102	103*	145	146	30
Labels per level	17-85	4-55-43-1	7-138	7-46-77-16	5-25
Average # characters	2,026	1,378	1,376	996	2,194
Train	18,692	23,149	31,306	58,715	266,666
Validation	4,674	n/a	6,262	14,785	66,667
Test	11,684	781,265	15,654	18,394	166,667

\* Overall, only 101 are available in training split.

\*\* Removing the unassigned categories.

## 5.1. Datasets used

In our experiments, we select three of the most popular datasets for HTC, namely the Web of Science (Kowsari et al., 2018), Blurb Genre Collection (Aly et al., 2019), and Reuters Corpus-V1 (Lewis et al., 2004). The latter two are distributed with pre-defined training and test splits, allowing us to directly compare results with other works. In order to diversify the domains being tested, we additionally test methods on a collection of bug reports that were crawled online, as well as a corpus of user reviews that we derive from the Amazon corpus. Overall, these datasets are representative of five different domains of applications of HTC methods (*i.e.*, books, scientific literature, news, IT tickets, and reviews), hence providing us with results across a wide spectrum of diverse data. Statistics for the five datasets are displayed in Table 6. The specific preprocessing procedure for each method we tested is discussed in the following sections, along with the experimental details.

### 5.1.1. Linux Bugs

The Linux Bugs dataset (which we will often refer to simply as “Bugs”) introduced by Lyubinetz et al. (2018) comprises bugs scraped from the Linux kernel bugtracker<sup>7</sup>. We utilize the script provided by the original authors to acquire a larger set of data. The documents are essentially support tickets classified in terms of importance, related product, and specific component. The “product” field acts as a parent label to the “component” sub-labels, from which

<sup>7</sup><https://bugzilla.kernel.org>

Listing 1: Linux bugs extracted from the Bugs dataset.

```

Description: "exact kernel version:linux-2.5.51 distribution:redhat 8.0+linux2.5.51 hardware
environment:intel stl2 mother boar d problem description: compile e100 as kernel module, insmod e100 and
start the network. then stop network and remove e100 together. then kernel crashes in random places. for
example: use command : insmod e100 /etc/init.d/network start /etc/init.d/network stop; rmmmod e100 then
the kernel crashes. eflags: 00010887 eip is at cascade+0x25/0x60 eax: defd02b8 ebx: 00000001 ecx:
00000000 edx: c150a4c0 esi: c150acd4 edi: c150acd4 ebp: c150acd4 esp: c0559f1c ds: 0068 es: 0068 ss:
0068 process swapper (pid: 0, threadinfo=c0558000 task=c0497f60) [...]"
Categories: "Drivers", "Networks"

```

we can derive a hierarchy of labels. As the depth of the hierarchy is only two, it is a rather shallow and wide structure. Furthermore, the dataset itself is strongly unbalanced; therefore, we discard bug reports tagged with labels or sub-labels appearing less than 100 times. The resulting dataset is still unbalanced, but this process helps to filter out lesser categories that are barely represented.

In terms of textual content, this dataset is very noisy; entries often contain grammatical inconsistencies and technical jargon, as well as technical readings such as stack traces or memory addresses. An example of a bug report from this dataset is given in Listing 1.

### 5.1.2. RCV1-v2

The Reuters Corpus Volume I (RCV1) dataset (Lewis et al., 2004) is a human-labeled newswire collection of Reuters News collected between August 20, 1996, and August 19, 1997. It contains over 800,000 manually categorized international newswire stories in English. In particular, we adopt the widely utilized corrected version, referred to as RCV1-v2. This version contains several fixes with respect to the categories assigned to each document, while 13 topic codes are removed entirely. We follow the instructions from Lewis et al. (2004) to generate the labeled dataset: all articles published from August 20, 1996, to August 31, 1996, are used as training split, while all the remaining articles (up to August 19, 1997) are used as test set. As a result of this chronological split, we noticed that 2 of the topic codes are only present in the testing set. We remove these codes, which leaves us with 101 topics in both training and testing. In our dataset, the headline and article fields of each news article are concatenated to generate the final document. The hierarchy among topic codes is specified in the file `rcv1.topics.hier.expanded` that can be downloaded in the official RCV1-v2 repository. An example of an article in XML extracted from RCV1-v2 is shown in Listing 2.

### 5.1.3. Web of Science

The Web Of Science (WOS) dataset, first introduced by Kowsari et al. (2017), contains abstracts from papers published on the Web of Science<sup>8</sup> platform. We utilize the dataset in its complete version (sometimes referred to as WOS-46985<sup>9</sup>), which comprises 46,985 abstracts from published papers in seven major scientific domains. The categories are further subdivided into 134 sub-domains. Much like the Linux Bugs dataset, it is characterized by a shallow hierarchy and an unnatural balancing — given by the fact that each example has exactly two labels (Aly et al., 2019). An example of an abstract from the WOS dataset is shown in Listing 3.

### 5.1.4. Blurb Genre Collection

The Blurb Genre Collection (BGC)<sup>10</sup> was first introduced by Aly et al. (2019) and is primarily comprised of so-called “blurbs” (*i.e.*, short advertising texts) in English, as well as other meta-data such as author name, publication date, and so on. The data is crawled from the Penguin Random House website and pre-processed to remove uninformative categories and category combinations that appear less than five times. As described by the authors, the procedure followed aims at mimicking the properties of the RCV1 dataset, ultimately generating a forest-like hierarchy structure. Label distribution is nonetheless unbalanced, with 146 overall labels and a hierarchy of depth 4. An example of a book summary extracted from this dataset is shown in Listing 4.

<sup>8</sup><https://www.webofscience.com>

<sup>9</sup><https://data.mendeley.com/datasets/9rw3vkcfy4/6>

<sup>10</sup><https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/blurb-genre-collection.html>

Listing 2: A news article extracted from the RCV1 dataset

```

<newsitem itemid="2307" id="root" date="1996-08-20" xml:lang="en">
  <title>UK: Oil prices slip as refiners shop for bargains.</title>
  <headline>Oil prices slip as refiners shop for bargains.</headline>
  <dateline>LONDON 1996-08-20</dateline>
  <text>
    <p>World oil prices slipped on Tuesday in a market where refiners stung by high crude premiums and poor margins began to bargain for a cheaper barrel.</p>
    <p>October futures for world benchmark Brent Blend crude from the North Sea closed down 38 cents at $20.43 a barrel after failing to break through the day's high of $20.80.</p>
    <p>&quot;There was a broad feeling in the market that Brent was overheated and had to come down,&quot; a trader said.</p>
    <p>On the unofficial Brent forward market, prompt differentials for Dated or physical Brent shrank, suggesting cargoes would fetch lower premiums in the weeks ahead. This could avert the risk of refineries running less crude through their systems to pump up the price of products.</p>
  [...]
  </text>
  <copyright>(c) Reuters Limited 1996</copyright>
  <metadata>
    <codes class="bip:countries:1.0"> [...] </codes>
    <codes class="bip:topics:1.0">
      <code code="M14"> [...] </code>
      <code code="M143"> [...] </code>
      <code code="MCAT"> [...] </code>
    </codes>
  </metadata>
</newsitem>

```

Listing 3: Abstract extracted from the WOS dataset

```

Abstract: "(T)his paper presents the concept of a software-defined radio with a flexible RF front end. The design and architecture of this system, as well as possible application examples will be explained. One specific scenario is the operation in maritime frequency bands. A well-known service is the Automatic Identification System (AIS), which has been captured by the DLR mission AISat, and will be chosen as a maritime application example. The results of an embedded solution for AIS on the SDR platform are presented in this paper. Since there is an increasing request for more performance on maritime radio bands, services like AIS will be enhanced by the International Association of Marine Aids to Navigation and Lighthouse Authorities (IALA). The new VHF Data Exchange Service (VDES) shall implement a dedicated satellite link. This paper describes that the SDR with a flexible RF front end can be used as a technology demonstration platform for this upcoming data exchange service."
Categories: "ECE", "Distributed-computing"

```

### 5.1.5. Amazon 5x5

We synthesize a new dataset composed of user reviews, labeled with the two-level category of the reviewed product. The dataset is obtained from the 2018 Amazon dump proposed in Ni et al. (2019). We extract product reviews for the following five categories: “Arts, Crafts and Sewing”, “Electronics”, “Grocery and Gourmet Food”, “Musical Instruments” and “Video Games”. Then, we manually exactly five sub-categories for each macro-category. In some cases, we map several sub-categories to the same macro-category (*e.g.*, “Needlework” and “Sewing” are related, hence grouped in “Sewing”) and fix inconsistencies as best as we can (*e.g.*, we list “Marshmallows” under the “Candy” category, instead of under the “Cooking & baking” one). Reviews are sampled in such a way that exactly 100,000 reviews are extracted for each domain. The dataset is therefore balanced concerning the 5 macro-categories. The second-level labels we obtain are also fairly balanced, though an equal split for each one is not ensured. Labels and sub-labels form a tree hierarchy, and there are exactly two labels for each sample. Reviews are the concatenation of a summary and a longer description, although sometimes only one of these is available. Before extraction, reviews are sorted by length, to ensure the longer ones are included in our dataset. A sample review extracted from this dataset is shown in Listing 5.

Listing 4: Book sample extracted from the BGC dataset

```

<book date="2018-08-18" xml:lang="en">
  <title>Creatures of the Night (Second Edition)</title>
  <body>Two of literary comics modern masters present a pair of magical and disturbing stories of strange
  creatures who are not quite what they seem! In The Price, a mysterious feline engages in a nightly
  conflict with an unseen, vicious foe. The Daughter of Owls recounts an eerie tale of a beautiful
  orphan girl who was found clutching an owl pellet-and how those who would do her wrong would face
  bizarre, unforeseen consequences. Neil Gaiman (The Sandman, American Gods) delivers his
  award-winning magic and mystery, realized in Michael Zulli's lavish paintings, newly re-designed in
  a beautiful new edition!</body>
  <copyright>(c) Penguin Random House</copyright>
  <metadata>
    <topics>
      <d0>Fiction</d0>
      <d1>Graphic Novels & Manga</d1>
    </topics>
    <author>Neil Gaiman</author>
    <published>Nov 29, 2016 </published>
    <page_num> 48 Pages</page_num>
    <isbn>9781506700250</isbn>
    <url> [...] </url>
  </metadata>
</book>

```

Listing 5: An example of user review in the Amazon dataset

```

{
  'summary': "Impressed for $10",
  'reviewText': "I picked up an old set of VW recaro bucket seats (the old gray ones). One of the cussions
  had a few holes in them (cigarette burn) so I decided I'd see what I could do. I worked on the seat
  in pieces so I did it inside. It could be done just as easily installed in the car. Simply use the
  color chart to mix a close match to the interior. Clean the area you will be working with. If
  necessary provide some backing such as a cotton ball stuffed down inside. [...] I applied the fix to
  several holes about 5 hours ago to writing this review and its already setting up nicely. I'm going
  to clear the excess coloring tomorrow and reassemble the seat and install them back in my car.
  Bottom line $10 and i could patch another 10 holes with moderately professional results. Can't beat
  it.",
  'category': ["Arts Crafts and Sewing", "Crafting"]
}

```

## 5.2. Models implemented

In this section, we provide some technical details on the methods we tested. From the pool of methods described in Section 4.2, we select as candidates the ones that also report results on the RCV1 and WOS datasets, since these are the most common datasets used in the surveyed works. These methods are listed in Table 7 in conjunction with the reported performance metrics. Among these, we further refine our selection by picking methods that have been used as baselines in previous works. Despite our best efforts, we were unable to use the existing implementations of many such methods. Firstly, several works have missing dependencies or files that are unspecified in the provided code. Secondly, some works utilize outdated versions of libraries that are no longer available or supported. Lastly, in other cases, there was a lack of instructions or support for the extension of the work on other datasets. We will briefly touch on the issue of reproducibility in Section 6; as mentioned, to help against this problema, we share publicly both our data splits (when legally possible) and code. In some cases, despite managing to run the method's implementation, we did not achieve good performance on any of our datasets. A more detailed recollection of our experiments on other methods may be found in the supplemental material.

The following paragraphs describe the hyper-parameters, preprocessing operations, and settings used with each method we were able to test, as well as a few baselines. For all methods, we perform 3-fold cross-validation on the Bugs dataset to select a common set of hyper-parameters. Then, we use these to test each model on all datasets. Obviously, custom fine-tuning could lead to better results for individual datasets. This is partly meant to reduce the

overall computational costs of the experimental part; at the same time, we do not claim to provide the best results possible for each method, but merely provide a comparison across both datasets and methods. Moreover, we aim to provide an outlook on the ease of applicability of these methods on diverse datasets. A supplementary material document is also provided to further detail the validation procedure.

*MATCH* We test the proposal from Zhang et al. (2021) on our datasets. Our datasets have no metadata; therefore, we test the model as limited to using the normalization mechanism for the hierarchy of labels. Additionally, we couldn't reproduce the joint embedding pre-training on our datasets, hence we replaced it with Word2Vec embeddings, which are previously fine-tuned on our dataset for 20 epochs. Before training Word2Vec embeddings, we preprocess the dataset by removing URLs, hexadecimal codes, and memory addresses (only for the Bugs dataset) and lowercase all words. The text is then tokenized with NLTK's word tokenizer, and finally punctuation marks are removed. The model uses a BCE loss with regularization terms, optimized using Adam.

*HiAGM* We use the public implementation of HiAGM (Zhou et al., 2020b) and test its performance on our datasets. Text is preprocessed with the same procedure described by its authors, consisting in lowercasing all words and removing special characters, stopwords, and punctuation marks. The Adam optimizer is used during training.

*BERT* We adopt the pre-trained "bert-base-cased" model from the HuggingFace library (Devlin et al., 2019; Wolf et al., 2020). BERT stacks several Transformer encoder blocks and we concatenate the (un-pooled) "[CLS]" representation from the last three blocks to obtain a more meaningful document representation, as was done in Marcuzzo et al. (2022b); Tanaka et al. (2019). As with MATCH, we preprocess the dataset by removing URLs, hexadecimal codes for the Bugs dataset, and lowercase words. For tokenization, we use BERT's Tokenizer, which is derived from WordPiece (Schuster and Nakajima, 2012 Mar 25–30). The BCE loss is adopted, and AdamW is utilized as optimizer.

*XML-CNN* We test a shallow CNN-based model from Liu et al. (2017); Adhikari et al. (2019). Word embeddings are initialized with pre-trained GloVe embeddings (Pennington et al., 2014), based on results obtained on the validation set. Text is preprocessed as was previously described for MATCH. Additionally, we remove stopwords from all corpora since we find this step improves metric results on the validation set. We use the BCE loss with the AdamW optimizer.

*CHAMP/MATCH losses* We additionally test the BERT model and XML-CNN with the CHAMP and MATCH loss proposed by Vaswani et al. (2022) and Zhang et al. (2021). For CHAMP, we implemented the "hard" version of their loss function. We test these approaches using the same optimization method, as well as the same preprocessing and tokenization steps described for BERT and XML-CNN, respectively.

*SVM* As a traditional baseline, we test the performance of simple SVM-based approaches with a one-vs-rest strategy. We test a standard flat approach, that performs multiclass classification on the most specific label for each example, as well as a naive multilabel approach with no hierarchical information (to simulate a NMLNP). We apply the same preprocessing described for XML-CNN and use TF-IDF representation to generate word feature vectors. As expected for a TF-IDF-based text representation, the removal of stopwords improves the performance on all datasets. On the Bugs dataset, we perform a more aggressive and targeted cleaning procedure, removing pieces of text that memory addresses and codes, such as to reduce the size of the generated vocabulary.

### 5.3. Results

In this section, we report and discuss results obtained with each method over the five datasets. As previously mentioned, the RCV1-v2 and BGC have been shared with pre-defined training and testing splits, allowing easy comparisons with other works. Results reported in the literature on the RCV1 are shown in Table 7 (only when using standard splits). The BGC dataset is relatively new and only a handful of works have results on it. Conversely, the WOS dataset, as well as the Amazon and Bugs dataset that we collected, do not have standardized splits. For this reason, we decide to share our training and testing splits, in a way to provide a benchmark for future works. Results over these splits are reported in Table 10. We also report results over these same datasets using a 3-fold CV strategy, with both standard and hierarchical metrics (when possible) in Tables 8 and 9.



Table 7

Performance metrics reported in literature over RCV1-v2 and WOS dataset

Method	RCV1-v2*		WOS**	
	micro- $F_1$	macro- $F_1$	micro- $F_1$	macro- $F_1$
Caps4HTC (Aly et al., 2019)	-	-	0.817	-
HTrans (Banerjee et al., 2019)	0.805	0.585	-	-
NeuralClassifier (RCNN) (Liu et al., 2019b)	0.810	0.533	-	-
HiLAP (Mao et al., 2019)	0.833	0.601	-	-
HiAGM-TP (Zhou et al., 2020b)	0.840	0.634	0.858	0.803
RLHR (Liu et al., 2021)	-	-	0.785	0.792
HCSM (Wang et al., 2021a)	0.858	0.609	0.921	0.807
HiMatch (Chen et al., 2021)	0.847	0.641	0.862	0.805
HIDDEN (Chatterjee et al., 2021)	0.793	0.473	-	-
HE-AGRCNN (Peng et al., 2021)	0.778	0.513	-	-
HVHMC (Xu et al., 2021)	-	-	0.743	-
SASF (Zhao et al., 2021)	-	-	0.867	0.811
HiDEC (Im et al., 2021)	0.844	0.623	0.806	-
HTCInfoMax (Deng et al., 2021)	0.835	0.627	0.856	0.800
PAMM-HiA-T5 (Huang et al., 2021)	-	-	0.904	0.816
HPT (Wang et al., 2022b)	0.873	0.695	0.872	0.819
HGCLR (Wang et al., 2022a)	0.865	0.683	0.871	0.812
Seq2Tree (Yu et al., 2022)	0.869	0.700	0.872	0.825
HBGL (Jiang et al., 2022)	0.872	0.711	0.874	0.820

\* Refers to RCV1-V2 with split 20,834/2,315/781,265 (train/validation/testing).

\*\* Refers to WOS-46985.

### 5.3.1. Comparison

On the Linux Bugs datasets, three methods perform particularly well: HiAGM, BERT (both base and with MATCH loss), and the SVM classifier. In terms of accuracy, the simplified SVM with multiclass objective achieves the highest result, though this could be considered unfair, as it makes certain assumptions based on the fact that the hierarchy is shallow and fixed (all targets are second-level leaves). Still, it is a decent contender for situations such as this. In general, the traditional approach using TF-IDF embeddings and the SVM classifier is likely effective in the Linux Bugs dataset because of how technical and noisy it is, which is both an advantage for its preprocessing steps as well as it being a disadvantage for context-aware methods such as BERT. The latter model achieves good results, but it is plausible to believe that it suffers from the lack of structure in the bodies of text within the dataset. On the other hand, HiAGM provides excellent results, with the highest score in terms of macro  $F_1$  score (though within a standard deviation of BERT). In some way, as Transformer-based approaches such as BERT are considered state-of-the-art on a wide array of NLP tasks in the literature, this result stands in favor of HiAGM, which can achieve comparable results by integrating hierarchical information within its framework. MATCH, XML-CNN, and the two hierarchical losses have provided worse or inconsequential results. Overall, the performance comparison seems to suggest that integrating hierarchical information is not straightforward but can be advantageous (such as in the case of HiAGM).

While the three top-performing methods remain the same, the Web of Science dataset largely favors the BERT-based approaches. The improvement provided by the MATCH loss is negligible, while the CHAMP loss seems to provide worse results (as was for the previous dataset). In general, then, the good results can be attributed to the semantic interpretation capabilities of BERT, rather than the hierarchical information that the former loss could provide. In this case, BERT surpasses the accuracy of traditional approaches based on SVMs, most likely because the text is much more structured and less noisy. HiAGM still provides decent results, though not as close to BERT as for the Linux Bugs dataset.

The Amazon 5x5 dataset synthesized for this work proved to be a much easier task for all the models, which may be attributed both to the fact that the hierarchy is very well-structured, as well as to the fact that the samples have been devised to be very balanced. Indeed, we found that during testing most methods would tend to entirely miss predictions of some low-frequency classes. This happened on all datasets except for Amazon 5x5, which, in that respect, is much “easier” because there are no real low-frequency classes. All methods bring decent results to the table,

**Table 8**

Performance metrics on the test set for each model and 3 datasets, with best results outlined in bold. (CL/ML=CHAMP/MATCH loss)

Model	Acc	F <sub>1</sub>	h-F <sub>1</sub>	AHC
<i>Bugs</i>				
MATCH	0.3624 [± 0.0491]	0.3635 [± 0.0559]	0.3638 [± 0.0560]	0.5501 [± 0.0834]
HiAGM	0.4482 [± 0.0069]	<b>0.5218 [± 0.0045]</b>	<b>0.5214 [± 0.0046]</b>	0.7672 [± 0.0527]
BERT + CL	0.4822 [± 0.0066]	0.4965 [± 0.0078]	0.4968 [± 0.0079]	0.4047 [± 0.0182]
BERT + ML	0.5061 [± 0.0099]	0.5165 [± 0.0095]	0.5172 [± 0.0096]	0.4634 [± 0.0238]
XML-CNN + CL	0.2603 [± 0.0037]	0.2515 [± 0.0062]	0.2522 [± 0.0247]	0.2443 [± 0.0052]
XML-CNN + ML	0.2823 [± 0.0110]	0.2767 [± 0.0058]	0.2774 [± 0.0057]	0.2885 [± 0.0188]
BERT	0.5070 [± 0.0113]	0.5204 [± 0.0093]	0.5209 [± 0.0091]	0.4606 [± 0.0119]
XML-CNN	0.2800 [± 0.0130]	0.2710 [± 0.0163]	0.2716 [± 0.0160]	0.3000 [± 0.0239]
SVM (MultiL)	0.3029 [± 0.0035]	0.4187 [± 0.0044]	0.4207 [± 0.0043]	0.5254 [± 0.0086]
SVM (MultiC)	<b>0.5496 [± 0.0039]</b>	0.4724 [± 0.0061]	-	-
<i>WOS</i>				
MATCH	0.5932 [± 0.0161]	0.6672 [± 0.0145]	0.6674 [± 0.0145]	0.3891 [± 0.0114]
HiAGM	0.6513 [± 0.0159]	0.7544 [± 0.0065]	0.7541 [± 0.0066]	0.4272 [± 0.0356]
BERT + CL	0.7647 [± 0.0064]	0.7928 [± 0.0066]	0.7929 [± 0.0066]	0.1941 [± 0.0130]
BERT + ML	<b>0.7718 [± 0.0027]</b>	0.7974 [± 0.0030]	0.7974 [± 0.0030]	0.2120 [± 0.0104]
XML-CNN + CL	0.4488 [± 0.0078]	0.5408 [± 0.0095]	0.5410 [± 0.0095]	0.1853 [± 0.0091]
XML-CNN + ML	0.4759 [± 0.0109]	0.5688 [± 0.0057]	0.5691 [± 0.0057]	0.2044 [± 0.0097]
BERT	0.7712 [± 0.0067]	<b>0.7981 [± 0.0059]</b>	<b>0.7981 [± 0.0059]</b>	0.2087 [± 0.0069]
XML-CNN	0.4714 [± 0.0115]	0.5628 [± 0.0085]	0.5630 [± 0.0085]	0.1978 [± 0.0105]
SVM (MultiL)	0.5051 [± 0.0026]	0.6611 [± 0.0018]	0.6619 [± 0.0018]	0.2172 [± 0.0061]
SVM (MultiC)	0.7609 [± 0.0033]	0.7397 [± 0.0045]	-	-
<i>Amazon</i>				
MATCH	0.8717 [± 0.0087]	0.9039 [± 0.0028]	0.9039 [± 0.0028]	0.0801 [± 0.0028]
HiAGM	0.8735 [± 0.0044]	0.9029 [± 0.0030]	0.9029 [± 0.0030]	0.0858 [± 0.0037]
BERT + CL	0.8912 [± 0.0019]	0.9192 [± 0.0015]	0.9192 [± 0.0015]	0.0632 [± 0.0016]
BERT + ML	<b>0.8960 [± 0.0021]</b>	<b>0.9214 [± 0.0015]</b>	<b>0.9214 [± 0.0015]</b>	0.0658 [± 0.0020]
XML-CNN + CL	0.8242 [± 0.0038]	0.8849 [± 0.0018]	0.8850 [± 0.0018]	0.0709 [± 0.0015]
XML-CNN + ML	0.8279 [± 0.0036]	0.8860 [± 0.0021]	0.8860 [± 0.0021]	0.0782 [± 0.0011]
BERT	0.8954 [± 0.0020]	0.9212 [± 0.0014]	0.9212 [± 0.0014]	0.0661 [± 0.0008]
XML-CNN	0.8290 [± 0.0030]	0.8867 [± 0.0016]	0.8867 [± 0.0017]	0.0774 [± 0.0020]
SVM (MultiL)	0.7340 [± 0.0016]	0.8491 [± 0.0010]	0.8492 [± 0.0010]	0.0628 [± 0.0010]
SVM (MultiC)	0.8666 [± 0.0007]	0.8677 [± 0.0007]	-	-

\* The standard deviation over the 2 repetitions of 3-fold CV is reported in brackets.

with the overall trend being similar to the one discussed for previous datasets: BERT performs very well, with a very minor improvement when adding MATCH's loss to its optimization. Other methods follow closely behind.

Among the ones discussed so far, the RCV1-v2 corpus is the first to have a deeper and more complex hierarchy. An interesting fact, then, is the fact that HiAGM performs best in terms of macro F<sub>1</sub> score on this dataset, further validating the injection of hierarchical information in HTC scenarios, as opposed to flat classification. The best-performing accuracy is achieved by the multiclass SVM, though it should be taken with a grain of salt because of how strong the flattening assumptions are on the relatively complex hierarchy. Indeed, this is reflected in its much lower F<sub>1</sub> score.

Lastly, the Blurb Genre Collection also proved quite difficult to categorize. Results are similar to the Web of Science dataset, with BERT-based models dominating in terms of F<sub>1</sub> score. Interestingly, the naive multilabel SVM approach seems to provide decent results in terms of this particular metric. Accuracy is yet again dominated by the multiclass SVM approach, though with a much worse F<sub>1</sub> score. HiAGM performs comparatively well, but, overall, lags behind BERT-based approaches on this dataset.

Overall, the multilabel BERT classifier with a flattened hierarchy achieved the best score across most datasets. The hierarchy-aware regularization strategies referred to as CHAMP and MATCH did not appear to consistently influence

**Table 9**

Performance metrics on the test set for the BGC and RCV1-v2 datasets, with best results outlined in bold.

Model	Acc	F <sub>1</sub>	h-F <sub>1</sub>	AHC
<i>RCV1-v2</i>				
MATCH	0.5149 [± 0.0010]	0.5308 [± 0.0056]	0.5309 [± 0.0056]	0.3478 [± 0.0005]
HiAGM	0.6035 [± 0.0064]	<b>0.6836 [± 0.0075]</b>	<b>0.6830 [± 0.0083]</b>	0.3236 [± 0.0001]
BERT + CL	0.6409 [± 0.0004]	0.6612 [± 0.0127]	0.6613 [± 0.0125]	0.1929 [± 0.0304]
BERT + ML	0.6383 [± 0.0059]	0.6805 [± 0.0073]	0.6806 [± 0.0073]	0.2270 [± 0.0076]
XML-CNN + CL	0.5449 [± 0.0099]	0.4898 [± 0.0058]	0.4905 [± 0.0055]	<b>0.1768 [± 0.0121]</b>
XML-CNN + ML	0.5460 [± 0.0089]	0.4832 [± 0.0161]	0.4840 [± 0.0164]	0.1917 [± 0.0156]
BERT	0.6391 [± 0.0036]	0.6722 [± 0.0305]	0.6723 [± 0.0304]	0.1959 [± 0.0480]
XML-CNN	0.5516 [± 0.0023]	0.4923 [± 0.0124]	0.4932 [± 0.0127]	0.1815 [± 0.0098]
SVM (MultiL)**	0.4971	0.5456	0.5472	0.2340
SVM (MultiC)**	<b>0.7289</b>	0.4416	-	-
<i>BGC</i>				
MATCH	0.3876 [± 0.0009]	0.4800 [± 0.0015]	0.4802 [± 0.0012]	0.4793 [± 0.0059]
HiAGM	0.4112 [± 0.0055]	0.5483 [± 0.0030]	0.5484 [± 0.0024]	0.5483 [± 0.0144]
BERT + CL	0.4674 [± 0.0008]	0.6058 [± 0.0156]	0.6060 [± 0.0155]	0.3303 [± 0.0017]
BERT + ML	0.4740 [± 0.0004]	<b>0.6116 [± 0.0049]</b>	<b>0.6119 [± 0.3450]</b>	0.3450 [± 0.0017]
XML-CNN + CL	0.3544 [± 0.0012]	0.3870 [± 0.0029]	0.3873 [± 0.0028]	0.3054 [± 0.0121]
XML-CNN + ML	0.3549 [± 0.0018]	0.3798 [± 0.0076]	0.3803 [± 0.0077]	<b>0.2875 [± 0.0227]</b>
BERT	0.4711 [± 0.0032]	0.6100 [± 0.0119]	0.6102 [± 0.0116]	0.3490 [± 0.0334]
XML-CNN	0.3602 [± 0.0028]	0.3996 [± 0.0092]	0.4000 [± 0.0091]	0.3096 [± 0.0073]
SVM (MultiL)**	0.3495	0.5129	0.5148	0.3985
SVM (MultiC)**	<b>0.6285</b>	0.2792	-	-

\* The standard deviation over 2 runs on the same standardized splits is reported in brackets.

\*\* SVMs run on fixed splits have deterministic results if run for enough iterations.

**Table 10**

 F<sub>1</sub> score (macro) results on one split of each dataset (avg of 2 runs)

Method	RCV1-v2	WOS	Amz	BGC	Bugs
MATCH	0.5308	0.6719	0.9048	0.4800	0.4064
HiAGM	<b>0.6836</b>	0.7484	0.9041	0.5483	<b>0.5263</b>
BERT + CL	0.6612	0.7958	0.9200	0.6058	0.5046
BERT + ML	0.6805	<b>0.8007</b>	<b>0.9214</b>	<b>0.6116</b>	0.5218
XML-CNN + CL	0.4898	0.5471	0.8856	0.3870	0.2517
XML-CNN + ML	0.4832	0.5645	0.8863	0.3798	0.2784
BERT	0.6722	0.7991	0.9206	0.6100	0.5235
XML-CNN	0.4923	0.5631	0.8855	0.3996	0.2802
SVM (MultiL)	0.5456	0.6610	0.8484	0.5129	0.4184
SVM (MultiC)	0.4416	0.7438	0.8676	0.2792	0.4700

results. At times they do provide a marginal boost in the precision or recall<sup>11</sup> over the flattened counterpart, but the advantage of using them is not evident as compared to BERT’s base model. The CNN-based model (XML-CNN) scored much lower than BERT across all datasets, and our experimental approach that integrated the aforementioned regularization strategies within its framework did not provide performance boosts.

Unfortunately, the evaluation based on hierarchical metrics does not provide any further insight into the benefits of the MATCH/CHAMP regularization. For the sake of transparency, we will underline the fact that we did not search for hyperparameters in a custom manner for each individual loss, but rather used the same that we found for the base models; it is possible that, with further tuning, clearer benefits could emerge. Moreover, since both approaches are essentially based on modified loss functions, different weighting coefficients could be used to strengthen or weaken

<sup>11</sup>Full results that include precision, recall, as well as their hierarchical counterparts, are available as supplemental material.

**Table 11**

Average of inference times over the Bugs dataset, averaged over all runs. Model variants are also averaged, as their inference time is comparable.

Method	Time (ms)
MATCH	$8.55 \times 10^{-3}$
HiAGM	$2.13 \times 10^{-1}$
SVM	$2.68 \times 10^{-2}$
BERT	$1.13 \times 10^1$
XML-CNN	$1.09 \times 10^{-2}$

the effect of the regularization, and should be fine-tuned specifically for each dataset. In our tests, we initialized these coefficients with the values suggested by the original authors, but a fine-tuning of these, paired with the models' own hyperparameters, could produce different results. We plan to focus more thoroughly on their applications in future work. Finally, in 10, we report results on a single split to enable easier comparisons with existing and future works that measure the performance on the same data. Briefly, HiAGM and BERT+ML appear to show the best results in terms of macro  $F_1$  score.

### 5.3.2. Inference time

In Table 11 we report the time required during inference for each model, using a single example (*i.e.*, batch size set to 1). Tests are carried out on an NVIDIA RTX 2080Ti GPU for all models that allow it. SVM-based models are run on CPU, on a separate machine running an Intel i7-8700. Machines have 64 and 48 GB of RAM, respectively.

The results in the table follow an interesting trend. The fastest method is MATCH, followed closely by the SVM-based and XML-CNN approaches. HiAGM is slightly slower, though not by a large magnitude. On the other hand, and expectedly, BERT has the longest inference time, two orders of magnitudes slower than HiAGM. Given these results, a real-time system would likely decide between the faster yet still reliable results of HiAGM and the slower BERT-based model. As mentioned, there is still room to integrate hierarchical information within Transformer-based approaches, therefore leading to stronger performances. Moreover, if the specific system has to deal with well-structured (syntactically and semantically) documents, BERT is likely to be the better choice.

## 5.4. Discussion

In the last few sections, we outlined the best-performing methods, the ones with faster inference time, and potential trade-offs. In this section, we summarize our thoughts on the experimental part of this survey.

Overall, Transformer-based models still shine in terms of performance and ease of use. This transfer learning technique is easy to adapt to new datasets and provides excellent results across the board. Its downsides come from its computational costs; training and fine-tuning these models are by far the most expensive process, and inference time is also the slowest across methods. Hence, their usage on real-time systems could be problematic, depending on the amount of data to process simultaneously. Traditional, SVM-based approaches applied naively can be a good alternative, though a more refined hierarchical approach could also be devised. There are many references in the literature on such approaches, yet we did not find a widespread and well-established implementation to use as a reference. Still, for systems that have to deal with noisy text, they are still valuable assets, as well as being generally easy to apply.

HiAGM, the state-of-the-art reference for many of the works we analyzed, works very well and brings points in favor of the injection of hierarchical information to improve the classification. In terms of usability, it was relatively straightforward to adapt, though some changes had to be made for it to be applicable to different hierarchies than the ones devised in the original work. Things were more complex for MATCH, as much of the training process was deeply embedded in the provided implementation, and had to be adapted to our datasets. This was the case for many other methods, and it is unfortunate that we were unable to test them because of inherent issues within them. Most would require re-writing them from scratch to be reproducible on our data, which would lead to a long and laborious process, falling outside the scope of our work.

Concerning hierarchical metrics, we did not find them to provide vastly interesting information on the results of the methods. The insight gained from these metrics would perhaps be more useful during the training process than it is

as an evaluation measure at the end of the process. Still, the concept behind penalizing “better mistakes” in a different manner is still interesting and warrants further investigation in the future.

## 6. Future work and research directions

There are a number of current challenges being tackled by current HTC research, providing for interesting research directions to explore. In this section, we briefly outline them.

First off, a considerable number of works on HTC have recently been taking into consideration the semantics of labels to improve classification performances (Chen et al., 2021, 2020; Xu et al., 2021; Ma et al., 2022). Some of them devise a way to obtain label embeddings and then use this information to produce label-aware document embeddings. This is different from standard classification approaches where document features are used to compute per-class probabilities, but an interesting direction being explored is the combination of the two. Some proposals even use contextualized language models to obtain embeddings for few-word labels. Despite this approach producing good results in (Chen et al., 2021), the usage of contextualized models to produce embeddings of small fragments of text without enough “context” is not completely justified according to previous literature (Gasparetto et al., 2022a), or at least it does not produce a proper semantic embedding. In a similar vein, some methods attempt to compensate for the lack of external metadata describing the label semantics by computing latent representation using auto-encoders (Ma et al., 2022) and GCNs (Peng et al., 2021; Xu et al., 2021).

Secondly, a worthwhile topic is that of reproducibility, benchmarks, and metrics. This is often a subject of debate in many sub-domains of machine learning, and rightfully so. First off, HTC lacks a proper set of benchmarks to allow for simpler comparison of methods and establishment of state-of-the-art. The wider TC branch of NLP, for instance, has multiple well-established benchmarks, such as GLUE (Wang et al., 2018 Nov) and SuperGLUE (Wang et al., 2019 Dec 8–14). As we outlined in Section 4.3, there is currently a wide range of datasets being utilized without much consistency. Even the most popular dataset, that of RCV1, is sometimes used inconsistently (as we outlined before, not all methods split the data in the same way). Overall, this makes reproducibility and comparability across methods much more difficult than they should be. Secondly, while we showcased the excellent efforts by many authors in the creation of hierarchical metrics (Sun and Lim, 2001; Silla and Freitas, 2011; Kosmopoulos et al., 2015; Stein et al., 2019), we also hinted at their flaws, as well as their lack of widespread adoption. Therefore, this is also still an open area of research. Lastly, as we showcased in this work, many proposed methods do not provide enough instructions to make their methods reproducible. For methods in Tables 1, 2, and 3, only about half provide a code implementation, and many of those still end up not being usable on our datasets because of a lack of instructions, poor dependencies, or excessive hardship in adapting it to other data. At the very least, a well-written set of instructions would be sufficient to provide enough information to any practitioner with the intent of reproducing a method.

Lastly, a topic that is drawing more and more attention across all fields of AI is that of *explainability*. Authors are exploring explainability in the broader NLP area by investigating LMs with tools such as Language Interpretability Tool (Tenney et al., 2020), Errudite (Wu et al., 2019), and iSEA (Yuan et al., 2022). This type of information could be utilized and refined to understand how an HTC method chooses its path through the hierarchy, and how it behaves when adjusting the input with small perturbations. Indeed, exploring the behavior of the model at different branches of the taxonomy could reveal interesting insights into the overall decision process.

## 7. Conclusions

In this article, we provide an overview of HTC approaches and the required background to fully grasp them, as well as an exploration of recent proposals in the field and in the broader NLP domain. We describe common frameworks utilized for HTC, as well as their strengths and weaknesses. We also present commonly utilized evaluation metrics, including the ones that are specifically designed to take into account the label taxonomy in their calculation. We then collect a number of works and proposals from recent years, and perform a more in-depth study on a selection of them, restricting ourselves to those methods that provide results on the most common datasets. Moreover, we provide an overview of commonly utilized datasets in HTC research. On the experimental side, we find that most of the methods analyzed do not provide an implementation, and, unfortunately, even many of those who do are not easily adjustable to different datasets, vastly limiting the practical value of these works. Thus, we proceed to measure the performance of those methods we manage to reproduce, using both traditional and hierarchical metrics, and comparing them with a set of baselines. Overall, our results suggest that integrating hierarchical information within the classification in HTC



datasets can indeed be useful, but is not trivial. The BERT-based baseline remained competitive (or better) than most approaches, though it must be said that there is still room to create customized BERT-based approaches for HTC. Finally, the code from our experiments is released, and our datasets are made available to researchers.

## CRedit authorship contribution statement

**Alessandro Zangari:** Conceptualization, Methodology, Investigation, Software, Data Curation, Writing - Original Draft. **Matteo Marcuzzo:** Conceptualization, Methodology, Software, Visualization, Writing - Original Draft. **Michele Schiavinato:** Conceptualization, Software, Investigation. **Matteo Rizzo:** Writing - Review & Editing, Validation. **Andrea Gasparetto:** Project administration, Methodology, Resources, Supervision, Writing - Review & Editing. **Andrea Albarelli:** Conceptualization, Methodology, Formal analysis, Validation, Supervision, Writing - Review & Editing.

## Acknowledgements

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## References

- Addi, H.A., Ezzahir, R., Mahmoudi, A., 2020. Three-level binary tree structure for sentiment classification in arabic text, in: Proceedings of the 3rd International Conference on Networking, Information Systems & Security, Association for Computing Machinery, New York, NY, USA. pp. 1–8. URL: <https://doi.org/10.1145/3386723.3387844>, doi:10.1145/3386723.3387844.
- Adhikari, A., Ram, A., Tang, R., Lin, J., 2019. DocBERT: BERT for document classification. arXiv (preprint) abs/1904.08398. doi:10.48550/arXiv.1904.08398, arXiv:1904.08398.
- Aho, A.V., Hopcroft, J.E., Ullman, J.D., 1976. On finding lowest common ancestors in trees. SIAM Journal on Computing 5, 115–132. URL: <https://doi.org/10.1137/0205011>, doi:10.1137/0205011, arXiv:<https://doi.org/10.1137/0205011>.
- Aljedani, N., Alotaibi, R., Taileb, M., 2021. Hmatc: Hierarchical multi-label arabic text classification model using machine learning. Egyptian Informatics Journal 22, 225–237. URL: <https://doi.org/10.1016/j.eij.2020.08.004>, doi:10.1016/j.eij.2020.08.004.
- Aly, R., Remus, S., Biemann, C., 2019. Hierarchical multi-label classification of text with capsule networks, in: Alva-Manchego, F., Choi, E., Khashabi, D. (Eds.), Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 2: Student Research Workshop, Association for Computational Linguistics. pp. 323–330. URL: <https://doi.org/10.18653/v1/p19-2045>, doi:10.18653/v1/p19-2045.
- Bahdanau, D., Cho, K., Bengio, Y., 2015. Neural machine translation by jointly learning to align and translate. arXiv abs/1409.0473. URL: <https://doi.org/10.48550/arXiv.1409.0473>, doi:10.48550/arXiv.1409.0473, arXiv:1409.0473.
- Banerjee, S., Akkaya, C., Perez-Sorrosal, F., Tsioutsouliklis, K., 2019. Hierarchical transfer learning for multi-label text classification, in: Korhonen, A., Traum, D.R., Márquez, L. (Eds.), Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers, Association for Computational Linguistics. pp. 6295–6300. URL: <https://doi.org/10.18653/v1/p19-1633>, doi:10.18653/v1/p19-1633.
- van den Bosch, A., 2017. Hidden Markov Models. Springer US, Boston, MA. chapter Hidden Markov Models. pp. 609–611. doi:10.1007/978-1-4899-7687-1\_124.
- Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers, in: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Association for Computing Machinery. pp. 144–152. doi:10.1145/130385.130401.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., 2020. Language models are few-shot learners, in: Advances in Neural Information Processing Systems, Curran Associates, Inc. pp. 1877–1901. URL: <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>.
- Caled, D., Won, M., Martins, B., Silva, M.J., 2019. A hierarchical label network for multi-label eurovoc classification of legislative contents, in: Doucet, A., Isaac, A., Golub, K., Aalberg, T., Jatowt, A. (Eds.), Digital Libraries for Open Knowledge, Springer International Publishing, Cham. pp. 238–252. doi:10.1007/978-3-030-30760-8\_21.
- Ceci, M., Malerba, D., 2007. Classifying web documents in a hierarchy of categories: a comprehensive study. Journal of Intelligent Information Systems 28, 37–78. URL: <https://doi.org/10.1007/s10844-006-0003-2>, doi:10.1007/s10844-006-0003-2.
- Cerri, R., Barros, R.C., de Carvalho, A.C.P.L.F., 2011. Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks, in: 2011 11th International Conference on Intelligent Systems Design and Applications, pp. 337–343. doi:10.1109/ISDA.2011.6121678.
- Cerri, R., Basgalupp, M.P., Barros, R.C., de Carvalho, A.C., 2019. Inducing hierarchical multi-label classification rules with genetic algorithms. Applied Soft Computing 77, 584–604. URL: <https://doi.org/10.1016/j.asoc.2019.01.017>, doi:10.1016/j.asoc.2019.01.017.
- Chatterjee, S., Maheshwari, A., Ramakrishnan, G., Jagarlapudi, S.N., 2021. Joint learning of hyperbolic label embeddings for hierarchical multi-label classification, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main

- Volume, Association for Computational Linguistics, Online. pp. 2829–2841. URL: <https://aclanthology.org/2021.eacl-main.247>, doi:10.18653/v1/2021.eacl-main.247.
- Chen, B., Huang, X., Xiao, L., Cai, Z., Jing, L., 2020. Hyperbolic interaction model for hierarchical multi-label classification. Proceedings of the AAAI Conference on Artificial Intelligence 34, 7496–7503. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/6247>, doi:10.1609/aaai.v34i05.6247.
- Chen, H., Ma, Q., Lin, Z., Yan, J., 2021. Hierarchy-aware label semantics matching network for hierarchical text classification, in: Zong, C., Xia, F., Li, W., Navigli, R. (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, Association for Computational Linguistics. pp. 4370–4379. URL: <https://doi.org/10.18653/v1/2021.acl-long.337>, doi:10.18653/v1/2021.acl-long.337.
- Cho, K., van Merriënboer, B., Bahdanau, D., Bengio, Y., 2014a. On the properties of neural machine translation: Encoder–decoder approaches, in: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Association for Computational Linguistics, Doha, Qatar. pp. 103–111. URL: <https://aclanthology.org/W14-4012>, doi:10.3115/v1/W14-4012.
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014b. Learning phrase representations using RNN encoder–decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics. pp. 1724–1734. doi:10.3115/v1/D14-1179.
- Ciapetti, A., Florio, R.D., Lomasto, L., Miscione, G., Ruggiero, G., Toti, D., 2019. NETHIC: A system for automatic text classification using neural networks and hierarchical taxonomies, in: Filipe, J., Smialek, M., Brodsky, A., Hammoudi, S. (Eds.), Proceedings of the 21st International Conference on Enterprise Information Systems, ICEIS 2019, Heraklion, Crete, Greece, May 3-5, 2019, Volume 1, SciTePress. pp. 296–306. URL: <https://doi.org/10.5220/0007709702960306>, doi:10.5220/0007709702960306.
- Cortes, C., Vapnik, V.N., 1995. Support-vector networks. Machine Learning 20. doi:10.1023/A:1022627411411.
- Deng, L., 2012. The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine 29, 141–142.
- Deng, Z., Peng, H., He, D., Li, J., Yu, P.S., 2021. HTCInfoMax: A global model for hierarchical text classification via information maximization, in: Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tür, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., Zhou, Y. (Eds.), Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, Association for Computational Linguistics. pp. 3259–3265. URL: <https://doi.org/10.18653/v1/2021.naacl-main.260>, doi:10.18653/v1/2021.naacl-main.260.
- Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. 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, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota. pp. 4171–4186. URL: <https://aclanthology.org/N19-1423>, doi:10.18653/v1/N19-1423.
- Dong, G., Zhang, W., Yadav, R., Mu, X., Zhou, Z., 2022. Owgc-hmc: An online web genre classification model based on hierarchical multilabel classification. Security and Communication Networks 2022, 7549880. URL: <https://doi.org/10.1155/2022/7549880>, doi:10.1155/2022/7549880.
- Dong, H., Wang, W., Huang, K., Coenen, F., 2021. Automated social text annotation with joint multilabel attention networks. IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 32, 2224–2238. doi:10.1109/TNNLS.2020.3002798.
- Elman, J.L., 1990. Finding structure in time. Cognitive Science 14, 179–211. URL: [https://doi.org/10.1207/s15516709cog1402\\_1](https://doi.org/10.1207/s15516709cog1402_1), doi:10.1207/s15516709cog1402\_1.
- Falis, M., Dong, H., Birch, A., Alex, B., 2021. Cophe: A count-preserving hierarchical evaluation metric in large-scale multi-label text classification, in: Moens, M., Huang, X., Specia, L., Yih, S.W. (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, Association for Computational Linguistics. pp. 907–912. URL: <https://doi.org/10.18653/v1/2021.emnlp-main.69>, doi:10.18653/v1/2021.emnlp-main.69.
- Fedus, W., Zoph, B., Shazeer, N., 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research 23, 1–39. URL: <http://jmlr.org/papers/v23/21-0998.html>, doi:10.48550/ARXIV.2101.03961.
- Ferdowsi, S., Borissov, N., Knafou, J., Amin, P., Teodoro, D., 2021. Classification of hierarchical text using geometric deep learning: the case of clinical trials corpus, in: Moens, M., Huang, X., Specia, L., Yih, S.W. (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, Association for Computational Linguistics. pp. 608–618. URL: <https://doi.org/10.18653/v1/2021.emnlp-main.48>, doi:10.18653/v1/2021.emnlp-main.48.
- Freitas, A., de Carvalho, A., 2007. A tutorial on hierarchical classification with applications in bioinformatics. Research and Trends in Data Mining Technologies and Applications doi:10.4018/978-1-59904-271-8.ch007.
- Gargiulo, F., Silvestri, S., Ciampi, M., Pietro, G.D., 2019. Deep neural network for hierarchical extreme multi-label text classification. Appl. Soft Comput. 79, 125–138. URL: <https://doi.org/10.1016/j.asoc.2019.03.041>, doi:10.1016/j.asoc.2019.03.041.
- Gasparetto, A., Marcuzzo, M., Zangari, A., Albarelli, A., 2022a. A survey on text classification algorithms: From text to predictions. Information 13. URL: <https://www.doi.org/10.3390/info13020083>, doi:10.3390/info13020083.
- Gasparetto, A., Zangari, A., Marcuzzo, M., Albarelli, A., 2022b. A survey on text classification: Practical perspectives on the Italian language. PLOS ONE 17, 1–46. URL: <https://doi.org/10.1371/journal.pone.0270904>, doi:10.1371/journal.pone.0270904.
- Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E., 2017. Neural message passing for quantum chemistry, in: Precup, D., Teh, Y.W. (Eds.), Proceedings of the 34th International Conference on Machine Learning, PMLR. pp. 1263–1272. URL: <https://proceedings.mlr.press/v70/gilmer17a.html>.
- Giunchiglia, E., Lukaszewicz, T., 2020. Coherent hierarchical multi-label classification networks, in: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 9662–9673. URL: <https://proceedings.neurips.cc/paper/2020/file/6dd4e10e3296fa63738371ec0d5df818-Paper.pdf>.

- Gong, J., Ma, H., Teng, Z., Teng, Q., Zhang, H., Du, L., Chen, S., Bhuiyan, M.Z.A., Li, J., Liu, M., 2020. Hierarchical graph transformer-based deep learning model for large-scale multi-label text classification. *IEEE Access* 8, 30885–30896. URL: <https://doi.org/10.1109/ACCESS.2020.2972751>, doi:10.1109/ACCESS.2020.2972751.
- Graves, A., 2013. Generating sequences with recurrent neural networks. *arXiv (preprint) abs/1308.0850*. URL: <https://doi.org/10.48550/arXiv.1308.0850>, doi:10.48550/arXiv.1308.0850, arXiv:1308.0850.
- Hinton, G.E., Krizhevsky, A., Wang, S.D., 2011. Transforming auto-encoders, in: Honkela, T., Duch, W., Girolami, M., Kaski, S. (Eds.), *Artificial Neural Networks and Machine Learning – ICANN 2011*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 44–51. URL: [https://doi.org/10.1007/978-3-642-21735-7\\_6](https://doi.org/10.1007/978-3-642-21735-7_6), doi:10.1007/978-3-642-21735-7\_6.
- Hinton, G.E., Sabour, S., Frosst, N., 2018. Matrix capsules with EM routing, in: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=HJWLfGWRb>.
- Ho, T.K., 1995. Random decision forests, in: *Proceedings of 3rd International Conference on Document Analysis and Recognition*, pp. 278–282. doi:10.1109/ICDAR.1995.598994.
- Hochreiter, S., Schmidhuber, J., 1997. Long Short-term Memory. *Neural computation* 9, 1735–1780. doi:10.1162/neco.1997.9.8.1735.
- Huang, W., Chen, E., Liu, Q., Chen, Y., Huang, Z., Liu, Y., Zhao, Z., Zhang, D., Wang, S., 2019. Hierarchical multi-label text classification: An attention-based recurrent network approach, in: Zhu, W., Tao, D., Cheng, X., Cui, P., Rundensteiner, E.A., Carmel, D., He, Q., Yu, J.X. (Eds.), *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, ACM, pp. 1051–1060. URL: <https://doi.org/10.1145/3357384.3357885>, doi:10.1145/3357384.3357885.
- Huang, W., Liu, C., Zhao, Y., Yang, X., Pan, Z., Zhang, Z., Liu, G., 2021. Hierarchy-aware T5 with path-adaptive mask mechanism for hierarchical text classification. *arXiv (preprint) abs/2109.08585*. URL: <https://arxiv.org/abs/2109.08585>, arXiv:2109.08585.
- Im, S., Kim, G., Oh, H., Jo, S., Kim, D., 2021. Hierarchy decoder is all you need to text classification. *arXiv (preprint) abs/2111.11104*. URL: <https://doi.org/10.48550/ARXIV.2111.11104>, doi:10.48550/ARXIV.2111.11104, arXiv:2111.11104.
- dbpedia.org [Internet], accessed 2022 Nov 15. DBpedia. <https://www.dbpedia.org>.
- wikimedia.org [Internet], accessed 2022 Nov 15. Wikimedia downloads. <https://dumps.wikimedia.org/backup-index.html>.
- Jawahar, G., Sagot, B., Seddah, D., 2019. What does BERT learn about the structure of language?, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy*, pp. 3651–3657. URL: <https://aclanthology.org/P19-1356>, doi:10.18653/v1/P19-1356.
- Jiang, H., Miao, Z., Lin, Y., Wang, C., Ni, M., Gao, J., Lu, J., Shi, G., 2020. Financial news annotation by weakly-supervised hierarchical multi-label learning, in: *Proceedings of the Second Workshop on Financial Technology and Natural Language Processing, -, Kyoto, Japan*, pp. 1–7. URL: <https://aclanthology.org/2020.finnlp-1.1>.
- Jiang, T., Wang, D., Sun, L., Chen, Z., Zhuang, F., Yang, Q., 2022. Exploiting global and local hierarchies for hierarchical text classification. *arXiv (preprint) URL: https://arxiv.org/abs/2205.02613*, doi:10.48550/ARXIV.2205.02613.
- Jones, K.S., 1972. A statistical interpretation of term specificity and its application in retrieval. *J. Doc.* 28, 11–21. doi:10.1108/eb026526.
- Jurafsky, D., Martin, J., 2020. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 3rd (draft) ed. URL: <https://web.stanford.edu/~jurafsky/slp3>.
- Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D., 2020. Scaling laws for neural language models. *arXiv (preprint) abs/2001.08361*. doi:10.48550/arXiv.2001.08361.
- Kim, Y., 2014. Convolutional neural networks for sentence classification, in: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 1746–1751. doi:10.3115/v1/D14-1181.
- Kiritchenko, S., Matwin, S., Nock, R., Famili, A.F., 2006. Learning and evaluation in the presence of class hierarchies: Application to text categorization, in: Lamontagne, L., Marchand, M. (Eds.), *Advances in Artificial Intelligence*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 395–406.
- Klimt, B., Yang, Y., 2004. The enron corpus: A new dataset for email classification research, in: *Proceedings of the 15th European Conference on Machine Learning*, Springer-Verlag, Berlin, Heidelberg, p. 217–226. URL: [https://doi.org/10.1007/978-3-540-30115-8\\_22](https://doi.org/10.1007/978-3-540-30115-8_22), doi:10.1007/978-3-540-30115-8\_22.
- Koller, D., Sahami, M., 1997. Hierarchically classifying documents using very few words, in: *Proceedings of the Fourteenth International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 170–178.
- Kosmopoulos, A., Partalas, I., Gaussier, E., Paliouras, G., Androutsopoulos, I., 2015. Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery* 29, 820–865. URL: <https://doi.org/10.1007/s10618-014-0382-x>, doi:10.1007/s10618-014-0382-x.
- Kowsari, K., Brown, D., Heidarysafa, M., Jafari Meimandi, K., Gerber, M., Barnes, L., 2018. Web of science dataset. doi:10.17632/9rw3vkcxfy4.6.
- Kowsari, K., Brown, D.E., Heidarysafa, M., Jafari Meimandi, K., Gerber, M.S., Barnes, L.E., 2017. HDLTex: Hierarchical deep learning for text classification, in: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 364–371. doi:10.1109/ICMLA.2017.0-134.
- Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D., 2019. Text classification algorithms: A survey. *Information* 10, doi:10.3390/info10040150.
- Kramer, M.A., 1991. Nonlinear principal component analysis using autoassociative neural networks. *AICHE Journal* 37, 233–243. URL: <https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690370209>, doi:https://doi.org/10.1002/aic.690370209, arXiv:https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690370209.
- Krendzelak, M., Jakab, F., 2020. Hierarchical text classification using CNNs with local approaches. *Comput. Informatics* 39, 907–924. URL: [https://doi.org/10.31577/cai\\_2020\\_5\\_907](https://doi.org/10.31577/cai_2020_5_907), doi:10.31577/cai\_2020\_5\_907.
- Kudo, T., Richardson, J., 2018 Oct 31 – Nov 4. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*,

- Association for Computational Linguistics, Brussels, Belgium. pp. 66–71. URL: <https://doi.org/10.18653/v1/D18-2012>, doi:10.18653/v1/D18-2012.
- Lang, K., 1995. Newsweeder: Learning to filter netnews, in: Prieditis, A., Russell, S. (Eds.), *Machine Learning Proceedings 1995*. Morgan Kaufmann, San Francisco (CA), pp. 331–339. URL: <https://www.sciencedirect.com/science/article/pii/B9781558603776500487>, doi:<https://doi.org/10.1016/B978-1-55860-377-6.50048-7>.
- Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D., 2017. Temporal convolutional networks for action segmentation and detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1003–1012. doi:10.1109/CVPR.2017.113.
- LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation* 1, 541–551. doi:10.1162/neco.1989.1.4.541.
- Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., Chen, Z., 2021 May 4. GShard: Scaling giant models with conditional computation and automatic sharding, in: *International Conference on Learning Representations (ICLR 2021)*, Vienna, Austria. doi:10.48550/arXiv.2006.16668.
- Lewis, D.D., Yang, Y., Rose, T.G., Li, F., 2004. Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.* 5, 361–397.
- Li, Q., Han, Z., Wu, X.M., 2018 Feb. Deeper insights into graph convolutional networks for semi-supervised learning, in: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI Press, New Orleans, Louisiana, USA. pp. 3538–3545.
- Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P.S., He, L., 2022. A survey on text classification: From traditional to deep learning. *ACM Trans. Intell. Syst. Technol.* 13. URL: <https://doi.org/10.1145/3495162>, doi:10.1145/3495162.
- Li, R.A., Hajjar, I., Goldstein, F., Choi, J.D., 2020. Analysis of hierarchical multi-content text classification model on B-SHARP dataset for early detection of alzheimer’s disease, in: Wong, K., Knight, K., Wu, H. (Eds.), *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2020*, Suzhou, China, December 4-7, 2020, Association for Computational Linguistics. pp. 358–365. URL: <https://aclanthology.org/2020.aacl-main.38/>.
- Li, X., Arora, K., Alaniyar, S., 2019. Mixed-model text classification framework considering the practical constraints, in: 2019 SECOND INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE FOR INDUSTRIES (AI4I 2019), IEEE; IEEE Comp Soc. IEEE, Laguna Hills, CA, USA. pp. 67–70. doi:10.1109/AI4I46381.2019.00024. 2nd IEEE International Conference on Artificial Intelligence for Industries (AI4I), Sep 25–27, 2019.
- Liang, X., Cheng, D., Yang, F., Luo, Y., Qian, W., Zhou, A., 2020. F-HMTC: Detecting financial events for investment decisions based on neural hierarchical multi-label text classification, in: Bessiere, C. (Ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, ijcai.org. pp. 4490–4496. URL: <https://doi.org/10.24963/ijcai.2020/619>, doi:10.24963/ijcai.2020/619.
- Liu, H., Zhang, D., Yin, B., Zhu, X., 2021. Improving pretrained models for zero-shot multi-label text classification through reinforced label hierarchy reasoning, in: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Online. pp. 1051–1062. URL: <https://aclanthology.org/2021.naacl-main.83>, doi:10.18653/v1/2021.naacl-main.83.
- Liu, J., Chang, W.C., Wu, Y., Yang, Y., 2017. Deep learning for extreme multi-label text classification, in: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Association for Computing Machinery. pp. 115–124. doi:10.1145/3077136.3080834.
- Liu, J., Xia, C., Yan, H., Xie, Z., Sun, J., 2019a. Hierarchical comprehensive context modeling for chinese text classification. *IEEE Access* 7, 154546–154559. URL: <https://doi.org/10.1109/ACCESS.2019.2949175>, doi:10.1109/ACCESS.2019.2949175.
- Liu, L., Mu, F., Li, P., Mu, X., Tang, J., Ai, X., Fu, R., Wang, L., Zhou, X., 2019b. Neuralclassifier: An open-source neural hierarchical multi-label text classification toolkit, in: Costa-jussà, M.R., Alfonseca, E. (Eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, Florence, Italy, July 28 - August 2, 2019, Volume 3: System Demonstrations, Association for Computational Linguistics. pp. 87–92. URL: <https://doi.org/10.18653/v1/p19-3015>, doi:10.18653/v1/p19-3015.
- Liu, P., Qiu, X., Huang, X., 2016. Recurrent neural network for text classification with multi-task learning, in: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, AAAI Press. p. 2873–2879.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., 2019c. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv abs/1907.11692*. doi:10.48550/arXiv.1907.11692, arXiv:1907.11692.
- Lomasto, L., Di Florio, R., Ciapetti, A., Miscione, G., Ruggiero, G., Toti, D., 2020. An automatic text classification method based on hierarchical taxonomies, neural networks and document embedding: The nethic tool, in: Filipe, J., Śmiałek, M., Brodsky, A., Hammoudi, S. (Eds.), *Enterprise Information Systems*, Springer International Publishing, Cham. pp. 57–77. URL: [https://doi.org/10.1007/978-3-030-40783-4\\_4](https://doi.org/10.1007/978-3-030-40783-4_4), doi:10.1007/978-3-030-40783-4\_4.
- Luong, T., Pham, H., Manning, C.D., 2015. Effective approaches to attention-based neural machine translation, in: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics. pp. 1412–1421. URL: <http://dx.doi.org/10.18653/v1/D15-1166>, doi:10.18653/v1/D15-1166.
- Lyubintsev, V., Boiko, T., Nicholas, D., 2018. Automated labeling of bugs and tickets using attention-based mechanisms in recurrent neural networks, in: 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), pp. 271–275. URL: <https://doi.org/10.1109/DSMP.2018.8478511>, doi:10.1109/DSMP.2018.8478511.
- Ma, Y., Liu, X., Zhao, L., Liang, Y., Zhang, P., Jin, B., 2022. Hybrid embedding-based text representation for hierarchical multi-label text classification. *Expert Syst. Appl.* 187, 115905. URL: <https://doi.org/10.1016/j.eswa.2021.115905>, doi:10.1016/j.eswa.2021.115905.
- Ma, Y., Zhao, J., Jin, B., 2020. A hierarchical fine-tuning approach based on joint embedding of words and parent categories for hierarchical multi-label text classification, in: Farkas, I., Masulli, P., Wermter, S. (Eds.), *Artificial Neural Networks and Machine Learning - ICANN 2020*



- 29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 15-18, 2020, Proceedings, Part II, Springer. pp. 746–757. URL: [https://doi.org/10.1007/978-3-030-61616-8\\_60](https://doi.org/10.1007/978-3-030-61616-8_60), doi:10.1007/978-3-030-61616-8\_60.
- Manning, C.D., Raghavan, P., Schütze, H., 2008a. Introduction to Information Retrieval. Cambridge University Press, USA.
- Manning, C.D., Raghavan, P., Schütze, H., 2008b. Introduction to Information Retrieval. Cambridge University Press, Cambridge, England, UK. doi:10.1017/CB09780511809071.
- Mao, Y., Tian, J., Han, J., Ren, X., 2019. Hierarchical text classification with reinforced label assignment, in: Intui, K., Jiang, J., Ng, V., Wan, X. (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, Association for Computational Linguistics. pp. 445–455. URL: <https://doi.org/10.18653/v1/D19-1042>, doi:10.18653/v1/D19-1042.
- Marcuzzo, M., Zangari, A., Albarelli, A., Gasparetto, A., 2022a. Recommendation systems: An insight into current development and future research challenges. IEEE Access 10, 86578–86623. doi:10.1109/ACCESS.2022.3194536.
- Marcuzzo, M., Zangari, A., Schiavinato, M., Giudice, L., Gasparetto, A., Albarelli, A., 2022b. A multi-level approach for hierarchical ticket classification, in: Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022), Association for Computational Linguistics, Gyeongju, Republic of Korea. pp. 201–214. URL: <https://aclanthology.org/2022.wnut-1.22>.
- Masmoudi, A., Bellaaj, H., Drira, K., Jmaiel, M., 2021. A co-training-based approach for the hierarchical multi-label classification of research papers. Expert Systems 38, e12613. URL: <https://doi.org/10.1111/exsy.12613>, doi:10.1111/exsy.12613.
- Masoudian, S., Derhami, V., Zarifzadeh, S., 2019. Hierarchical persian text categorization in absence of labeled data, in: 2019 27th Iranian Conference on Electrical Engineering (ICEE), pp. 1951–1955. doi:10.1109/IranianCEE.2019.8786690.
- McAuley, J., Leskovec, J., 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text, in: Proceedings of the 7th ACM Conference on Recommender Systems, Association for Computing Machinery, New York, NY, USA. p. 165–172. URL: <https://doi.org/10.1145/2507157.2507163>, doi:10.1145/2507157.2507163.
- Meng, Y., Shen, J., Zhang, C., Han, J., 2019. Weakly-supervised hierarchical text classification, in: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, AAAI Press. pp. 6826–6833. URL: <https://doi.org/10.1609/aaai.v33i01.33016826>, doi:10.1609/aaai.v33i01.33016826.
- Mielke, S.J., Alyafeai, Z., Salesky, E., Raffel, C., Dey, M., Gallé, M., Raja, A., Si, C., Lee, W.Y., Sagot, B., Tan, S., 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. arXiv abs/2112.10508. arXiv:2112.10508.
- Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013a. Efficient estimation of word representations in vector space, in: 1st International Conference on Learning Representations, ICLR 2013. doi:10.48550/arXiv.1301.3781, arXiv:1301.3781.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J., 2013b. Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems 26, Curran Associates, Inc.. pp. 3111–3119. URL: <https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf>.
- Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., Gao, J., 2021. Deep learning–based text classification: A comprehensive review. Acm Comput. Surv. 54. doi:10.1145/3439726.
- Nakano, F.K., Cerri, R., Vens, C., 2020. Active learning for hierarchical multi-label classification. Data Mining and Knowledge Discovery 34, 1496–1530. URL: <https://doi.org/10.1007/s10618-020-00704-w>, doi:10.1007/s10618-020-00704-w.
- Ni, J., Li, J., McAuley, J., 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China. pp. 188–197. URL: <https://aclanthology.org/D19-1018>, doi:10.18653/v1/D19-1018.
- Pal, A., Selvakumar, M., Sankarasubbu, M., 2020. Magnet: Multi-label text classification using attention-based graph neural network, in: Proceedings of the 12th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART, INSTICC. SciTePress. pp. 494–505. doi:10.5220/0008940304940505.
- Pascanu, R., Mikolov, T., Bengio, Y., 2013. On the difficulty of training recurrent neural networks, in: Dasgupta, S., McAllester, D. (Eds.), Proceedings of the 30th International Conference on Machine Learning, PMLR, Atlanta, Georgia, USA. pp. 1310–1318. URL: <https://proceedings.mlr.press/v28/pascanu13.html>.
- Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., Song, Y., Yang, Q., 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn, in: Proceedings of the 2018 World Wide Web Conference, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE. p. 1063–1072. URL: <https://doi.org/10.1145/3178876.3186005>, doi:10.1145/3178876.3186005.
- Peng, H., Li, J., Wang, S., Wang, L., Gong, Q., Yang, R., Li, B., Yu, P.S., He, L., 2021. Hierarchical taxonomy-aware and attentional graph capsule rnns for large-scale multi-label text classification. IEEE Trans. Knowl. Data Eng. 33, 2505–2519. URL: <https://doi.org/10.1109/TKDE.2019.2959991>, doi:10.1109/TKDE.2019.2959991.
- Pennington, J., Socher, R., Manning, C., 2014. GloVe: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics. pp. 1532–1543. doi:10.3115/v1/D14-1162.
- Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L., 2018 Jun 1–6. Deep contextualized word representations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana. pp. 2227–2237. doi:10.18653/v1/N18-1202.
- Prabowo, F.A., Ibrohim, M.O., Budi, I., 2019. Hierarchical multi-label classification to identify hate speech and abusive language on Indonesian twitter, in: 2019 6th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE), pp. 1–5. doi:10.1109/ICITACEE.2019.8904425.



- Pujari, S.C., Friedrich, A., Strötgen, J., 2021. A multi-task approach to neural multi-label hierarchical patent classification using transformers, in: Hiemstra, D., Moens, M.F., Mothe, J., Perego, R., Potthast, M., Sebastiani, F. (Eds.), *Advances in Information Retrieval*, Springer International Publishing, Cham. pp. 513–528.
- Punera, K., Ghosh, J., 2008. Enhanced hierarchical classification via isotonic smoothing, in: *Proceedings of the 17th International Conference on World Wide Web*, Association for Computing Machinery, New York, NY, USA. p. 151–160. URL: <https://doi.org/10.1145/1367497.1367518>, doi:10.1145/1367497.1367518.
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., 2018. Improving language understanding by generative pre-training. URL: [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf).
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., 2019. Language models are unsupervised multitask learners. URL: [https://cdn.openai.com/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf).
- Rojas, K.R., Bustamante, G., Oncevay, A., Cabezudo, M.A.S., 2020. Efficient strategies for hierarchical text classification: External knowledge and auxiliary tasks, in: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Association for Computational Linguistics. pp. 2252–2257. URL: <https://doi.org/10.18653/v1/2020.acl-main.205>, doi:10.18653/v1/2020.acl-main.205.
- Romero, M., Finke, J., Rocha, C., 2022. A top-down supervised learning approach to hierarchical multi-label classification in networks. *Applied Network Science* 7, 8. URL: <https://doi.org/10.1007/s41109-022-00445-3>, doi:10.1007/s41109-022-00445-3.
- Rumelhart, D.E., Hinton, G.E., Williams, R.J., Jan 1986. *Learning Internal Representations by Error Propagation*. MIT Press, Cambridge, MA, USA. chapter 11. pp. 318–362.
- Sabour, S., Frosst, N., Hinton, G.E., 2017. Dynamic routing between capsules, in: *Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA*. pp. 3859–3869.
- Safavian, S., Landgrebe, D., 1991. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics* 21, 660–674. doi:10.1109/21.97458.
- Sainte Fare Garnot, V., Landrieu, L., 2021. Leveraging class hierarchies with metric-guided prototype learning, in: *32th British Machine Vision Conference, BMVA Press, Online*. URL: <https://arxiv.org/abs/2007.03047>.
- Salesky, E., Etter, D., Post, M., 2021 Nov. Robust open-vocabulary translation from visual text representations, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic*. pp. 7235–7252. URL: <https://aclanthology.org/2021.emnlp-main.576>, doi:10.18653/v1/2021.emnlp-main.576.
- Sandhaus, E., 2008. *The New York Times Annotated Corpus LDC2008T19*. <https://doi.org/10.35111/77ba-9x74>. Philadelphia: Linguistic Data Consortium.
- Schuster, M., Nakajima, K., 2012 Mar 25–30. Japanese and korean voice search, in: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan. pp. 5149–5152. doi:10.1109/ICASSP.2012.6289079.
- Sebastiani, F., 2002. Machine learning in automated text categorization. *ACM Comput. Surv.* 34, 1–47. URL: <https://doi.org/10.1145/505282.505283>, doi:10.1145/505282.505283.
- Sennrich, R., Haddow, B., Birch, A., 2016 Aug 7–12. Neural machine translation of rare words with subword units, in: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Berlin, Germany. pp. 1715–1725. URL: <https://doi.org/10.18653/v1/P16-1162>, doi:10.18653/v1/P16-1162.
- Shen, J., Qiu, W., Meng, Y., Shang, J., Ren, X., Han, J., 2021. Taxoclass: Hierarchical multi-label text classification using only class names, in: Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tür, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., Zhou, Y. (Eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, Association for Computational Linguistics. pp. 4239–4249. URL: <https://doi.org/10.18653/v1/2021.naacl-main.335>, doi:10.18653/v1/2021.naacl-main.335.
- Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P., 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* 30, 83–98. doi:10.1109/MSP.2012.2235192.
- Silla, C.N., Freitas, A.A., 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22, 31–72. URL: <https://doi.org/10.1007/s10618-010-0175-9>, doi:10.1007/s10618-010-0175-9.
- da Silva, L.V.M., Cerri, R., 2021. Feature selection for hierarchical multi-label classification, in: Abreu, P.H., Rodrigues, P.P., Fernández, A., Gama, J. (Eds.), *Advances in Intelligent Data Analysis XIX*, Springer International Publishing, Cham. pp. 196–208. doi:10.1007/978-3-030-74251-5\_16.
- Song, Y., Yan, Z., Qin, Y., Zhao, D., Ye, X., Chai, Y., Ouyang, Y., 2022. Hierarchical multi-label text classification based on a matrix factorization and recursive-attention approach, in: *2022 7th International Conference on Big Data Analytics (ICBDA)*, pp. 170–176. doi:10.1109/ICBDA55095.2022.9760305.
- Stein, R.A., Jaques, P.A., Valiati, J.F., 2019. An analysis of hierarchical text classification using word embeddings. *Inf. Sci.* 471, 216–232. URL: <https://doi.org/10.1016/j.ins.2018.09.001>, doi:10.1016/j.ins.2018.09.001.
- Stepišnik, T., Kocev, D., 2020. Hyperbolic embeddings for hierarchical multi-label classification, in: Helic, D., Leitner, G., Stettinger, M., Felfernig, A., Raš, Z.W. (Eds.), *Foundations of Intelligent Systems*, Springer International Publishing, Cham. pp. 66–76. doi:10.1007/978-3-030-59491-6\_7.
- Sun, A., Lim, E.P., 2001. Hierarchical text classification and evaluation, in: *Proceedings of the 2001 IEEE International Conference on Data Mining*, IEEE Computer Society, USA. p. 521–528.
- Sun, A., Lim, E.P., Ng, W.K., 2003. *Hierarchical Text Classification Methods and Their Specification*. Springer US, Boston, MA. chapter 14. pp. 236–256. URL: [https://doi.org/10.1007/978-1-4615-0435-1\\_14](https://doi.org/10.1007/978-1-4615-0435-1_14), doi:10.1007/978-1-4615-0435-1\_14.
- Sun, A., Lim, E.P., Ng, W.K., Srivastava, J., 2004. Blocking reduction strategies in hierarchical text classification. *IEEE Transactions on Knowledge and Data Engineering* 16, 1305–1308. doi:10.1109/TKDE.2004.50.

- Sutskever, I., Vinyals, O., Le, Q.V., 2014 Dec 8–13. Sequence to sequence learning with neural networks, in: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, MIT Press, Cambridge, MA, USA. pp. 3104–3112.
- Tanaka, H., Shinou, H., Cao, R., Bai, J., Ma, W., 2019. Document classification by word embeddings of BERT, in: Nguyen, L.M., Phan, X.H., Hasida, K., Tojo, S. (Eds.), 16th International Conference of the Pacific Association for Computational Linguistics, PACLING 2019, Springer Singapore, Hanoi, Vietnam. pp. 145–154. doi:10.1007/978-981-15-6168-9\_13.
- Tenney, I., Wexler, J., Bastings, J., Bolukbasi, T., Coenen, A., Gehrmann, S., Jiang, E., Pushkarna, M., Radebaugh, C., Reif, E., Yuan, A., 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online. pp. 107–118. URL: <https://doi.org/10.18653/v1/2020.emnlp-demos.15>, doi:10.18653/v1/2020.emnlp-demos.15.
- Tsai, S.C., Huang, C.W., Chen, Y.N., 2021. Modeling diagnostic label correlation for automatic ICD coding, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online. pp. 4043–4052. URL: <https://aclanthology.org/2021.naacl-main.318>, doi:10.18653/v1/2021.naacl-main.318.
- Vaswani, A., Aggarwal, G., Netrapalli, P., Hegde, N.G., 2022. All mistakes are not equal: Comprehensive hierarchy aware multi-label predictions (champ). arXiv (preprint) URL: <https://doi.org/10.48550/ARXIV.2206.08653>, doi:10.48550/ARXIV.2206.08653.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc.. pp. 6000–6010.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks, in: International Conference on Learning Representations. URL: <https://openreview.net/forum?id=rJXMpikCZ>.
- Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H., 2008. Decision trees for hierarchical multi-label classification. Machine Learning 73, 185. URL: <https://doi.org/10.1007/s10994-008-5077-3>, doi:10.1007/s10994-008-5077-3.
- de Vries, W., van Cranenburgh, A., Nissim, M., 2020. What’s so special about BERT’s layers? a closer look at the NLP pipeline in monolingual and multilingual models, in: Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online. pp. 4339–4350. URL: <https://aclanthology.org/2020.findings-emnlp.389>, doi:10.18653/v1/2020.findings-emnlp.389.
- Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R., 2019 Dec 8–14. Superglue: A stickier benchmark for general-purpose language understanding systems, in: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Curran Associates Inc., Vancouver, Canada. pp. 3266–3280.
- Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S., 2018 Nov. GLUE: A multi-task benchmark and analysis platform for natural language understanding, in: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, Brussels, Belgium. pp. 353–355. doi:10.18653/v1/W18-5446.
- Wang, B., Hu, X., Li, P., Yu, P.S., 2021a. Cognitive structure learning model for hierarchical multi-label text classification. Knowl. Based Syst. 218, 106876. URL: <https://doi.org/10.1016/j.knsys.2021.106876>, doi:10.1016/j.knsys.2021.106876.
- Wang, D., Cui, P., Zhu, W., 2016. Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA. pp. 1225–1234. URL: <https://doi.org/10.1145/2939672.2939753>, doi:10.1145/2939672.2939753.
- Wang, X., Zhao, L., Liu, B., Chen, T., Zhang, F., Wang, D., 2021b. Concept-based label embedding via dynamic routing for hierarchical text classification, in: Zong, C., Xia, F., Li, W., Navigli, R. (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, Association for Computational Linguistics. pp. 5010–5019. URL: <https://doi.org/10.18653/v1/2021.acl-long.388>, doi:10.18653/v1/2021.acl-long.388.
- Wang, Z., Wang, P., Huang, L., Sun, X., Wang, H., 2022a. Incorporating hierarchy into text encoder: a contrastive learning approach for hierarchical text classification, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland. pp. 7109–7119. URL: <https://aclanthology.org/2022.acl-long.491>, doi:10.18653/v1/2022.acl-long.491.
- Wang, Z., Wang, P., Liu, T., Cao, Y., Sui, Z., Wang, H., 2022b. HPT: hierarchy-aware prompt tuning for hierarchical text classification. arXiv URL: <https://arxiv.org/abs/2204.13413>, doi:10.48550/ARXIV.2204.13413.
- Wehrmann, J., Cerri, R., Barros, R., 2018. Hierarchical multi-label classification networks, in: Dy, J., Krause, A. (Eds.), Proceedings of the 35th International Conference on Machine Learning, PMLR. pp. 5075–5084. URL: <https://proceedings.mlr.press/v80/wehrmann18a.html>.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., Rush, A., 2020. Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics. pp. 38–45. doi:10.18653/v1/2020.emnlp-demos.6.
- Wu, T., Ribeiro, M.T., Heer, J., Weld, D., 2019. Errudite: Scalable, reproducible, and testable error analysis, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy. pp. 747–763. URL: <https://aclanthology.org/P19-1073>, doi:10.18653/v1/P19-1073.
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S., 2021. A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 32, 4–24. doi:10.1109/TNNLS.2020.2978386.
- Xi, E., Bing, S., Jin, Y., 2017. Capsule Network Performance on Complex Data. arXiv (preprint) URL: <https://doi.org/10.48550/ARXIV.1712.03480>, doi:10.48550/ARXIV.1712.03480.
- Xiao, H., Liu, X., Song, Y., 2019. Efficient path prediction for semi-supervised and weakly supervised hierarchical text classification, in: Liu, L., White, R.W., Mantrach, A., Silvestri, F., McAuley, J.J., Baeza-Yates, R., Zia, L. (Eds.), The World Wide Web Conference, WWW 2019,

- San Francisco, CA, USA, May 13-17, 2019, ACM. pp. 3370–3376. URL: <https://doi.org/10.1145/3308558.3313658>, doi:10.1145/3308558.3313658.
- Xu, J., Du, Q., 2020. Learning neural networks for text classification by exploiting label relations. *Multimedia Tools and Applications* 79, 22551–22567. URL: <https://doi.org/10.1007/s11042-020-09063-6>, doi:10.1007/s11042-020-09063-6.
- Xu, L., Teng, S., Zhao, R., Guo, J., Xiao, C., Jiang, D., Ren, B., 2021. Hierarchical multi-label text classification with horizontal and vertical category correlations, in: Moens, M., Huang, X., Specia, L., Yih, S.W. (Eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Association for Computational Linguistics. pp. 2459–2468. URL: <https://doi.org/10.18653/v1/2021.emnlp-main.190>, doi:10.18653/v1/2021.emnlp-main.190.
- Xu, S., Li, Y., Wang, Z., 2017. Bayesian multinomial naïve bayes classifier to text classification, in: *Advanced Multimedia and Ubiquitous Engineering*, Springer Singapore. pp. 347–352. doi:10.1007/978-981-10-5041-1\_57.
- Xu, Z., Zhang, B., Li, D., Yue, X., 2022. Hierarchical multilabel classification by exploiting label correlations. *International Journal of Machine Learning and Cybernetics* 13, 115–131. URL: <https://doi.org/10.1007/s13042-021-01371-z>, doi:10.1007/s13042-021-01371-z.
- Xun, G., Jha, K., Sun, J., Zhang, A., 2020. Correlation networks for extreme multi-label text classification, in: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Association for Computing Machinery, New York, NY, USA. pp. 1074–1082. URL: <https://doi.org/10.1145/3394486.3403151>, doi:10.1145/3394486.3403151.
- Yan, J., Mu, L., Wang, L., Ranjan, R., Zomaya, A.Y., 2020. Temporal convolutional networks for the advance prediction of enso. *Scientific Reports* 10, 8055. doi:10.1038/s41598-020-65070-5.
- Yang, P., Sun, X., Li, W., Ma, S., Wu, W., Wang, H., 2018. SGM: Sequence generation model for multi-label classification, in: *Proceedings of the 27th International Conference on Computational Linguistics*, Association for Computational Linguistics, Santa Fe, New Mexico, USA. pp. 3915–3926. URL: <https://aclanthology.org/C18-1330>.
- Yang, Y., Wang, H., Zhu, J., Shi, W., Guo, W., Zhang, J., 2021. Effective seed-guided topic labeling for dataless hierarchical short text classification, in: *Brambilla, M., Chbeir, R., Frasinca, F., Manolescu, I. (Eds.), Web Engineering - 21st International Conference, ICWE 2021, Biarritz, France, May 18-21, 2021*, Proceedings, Springer. pp. 271–285. URL: [https://doi.org/10.1007/978-3-030-74296-6\\_21](https://doi.org/10.1007/978-3-030-74296-6_21), doi:10.1007/978-3-030-74296-6\_21.
- Yang, Z., Liu, G., 2019. Hierarchical sequence-to-sequence model for multi-label text classification. *IEEE Access* 7, 153012–153020. URL: <https://doi.org/10.1109/ACCESS.2019.2948855>, doi:10.1109/ACCESS.2019.2948855.
- Yao, L., Mao, C., Luo, Y., 2019. Graph convolutional networks for text classification. *Proc. AAAI Conf. Artif. Intell.* 33, 7370–7377. doi:10.1609/aaai.v33i01.33017370.
- Ye, C., Zhang, L., He, Y., Zhou, D., Wu, J., 2021. Beyond text: Incorporating metadata and label structure for multi-label document classification using heterogeneous graphs, in: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic. pp. 3162–3171. URL: <https://aclanthology.org/2021.emnlp-main.253>, doi:10.18653/v1/2021.emnlp-main.253.
- Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.Y., 2021. Do transformers really perform badly for graph representation?, in: *Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (Eds.), Advances in Neural Information Processing Systems*, Curran Associates, Inc.. pp. 28877–28888. URL: <https://proceedings.neurips.cc/paper/2021/file/f1c1592588411002af340cbaedd6fc33-Paper.pdf>.
- Yu, C., Shen, Y., Mao, Y., 2022. Constrained sequence-to-tree generation for hierarchical text classification, in: *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Association for Computing Machinery, New York, NY, USA. pp. 1865–1869. URL: <https://doi.org/10.1145/3477495.3531765>, doi:10.1145/3477495.3531765.
- Yu, Y., Sun, Z., Sun, C., Liu, W., 2021. Hierarchical multilabel text classification via multitask learning, in: *33rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2021, Washington, DC, USA, November 1-3, 2021*, IEEE. pp. 1138–1143. URL: <https://doi.org/10.1109/ICTAI52525.2021.00180>, doi:10.1109/ICTAI52525.2021.00180.
- Yuan, J., Vig, J., Rajani, N., 2022. Isea: An interactive pipeline for semantic error analysis of nlp models, in: *27th International Conference on Intelligent User Interfaces*, Association for Computing Machinery, New York, NY, USA. pp. 878–888. URL: <https://doi.org/10.1145/3490099.3511146>, doi:10.1145/3490099.3511146.
- Zangari, A., Marcuzzo, M., Schiavinato, M., Albarelli, A., Gasparetto, A., Rizzo, M., 2022. [dataset] Hierarchical Text Classification corpora (v.1), Zenodo.org. URL: <https://doi.org/10.5281/zenodo.7319519>, doi:10.5281/zenodo.7319519.
- Zhang, M.L., Zhou, Z.H., 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26, 1819–1837. doi:10.1109/TKDE.2013.39.
- Zhang, X., Xu, J., Soh, C., Chen, L., 2022. LA-HCN: label-based attention for hierarchical multi-label text classification neural network. *Expert Syst. Appl.* 187, 115922. URL: <https://doi.org/10.1016/j.eswa.2021.115922>, doi:10.1016/j.eswa.2021.115922.
- Zhang, Y., Shen, Z., Dong, Y., Wang, K., Han, J., 2021. Match: Metadata-aware text classification in a large hierarchy, in: *Proceedings of the Web Conference 2021*, Association for Computing Machinery, New York, NY, USA. pp. 3246–3257. URL: <https://doi.org/10.1145/3442381.3449979>, doi:10.1145/3442381.3449979.
- Zhang, Y., Wallace, B.C., 2017. A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification, in: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (IJCNLP)*, Asian Federation of Natural Language Processing, Taipei, Taiwan. pp. 253–263.
- Zhao, R., Wei, X., Ding, C., Chen, Y., 2021. Hierarchical multi-label text classification: Self-adaption semantic awareness network integrating text topic and label level information, in: *Qiu, H., Zhang, C., Fei, Z., Qiu, M., Kung, S.Y. (Eds.), Knowledge Science, Engineering and Management*, Springer International Publishing, Cham. pp. 406–418. URL: [https://doi.org/10.1007/978-3-030-82147-0\\_33](https://doi.org/10.1007/978-3-030-82147-0_33), doi:10.1007/978-3-030-82147-0\_33.

- Zhao, W., Gao, H., Chen, S., Wang, N., 2020. Generative multi-task learning for text classification. *IEEE Access* 8, 86380–86387. doi:10.1109/ACCESS.2020.2991337.
- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020a. Graph neural networks: A review of methods and applications. *AI Open* 1, 57–81. URL: <https://www.sciencedirect.com/science/article/pii/S2666651021000012>, doi:<https://doi.org/10.1016/j.aiopen.2021.01.001>.
- Zhou, J., Ma, C., Long, D., Xu, G., Ding, N., Zhang, H., Xie, P., Liu, G., 2020b. Hierarchy-aware global model for hierarchical text classification, in: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (Eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, Association for Computational Linguistics*. pp. 1106–1117. URL: <https://doi.org/10.18653/v1/2020.acl-main.104>, doi:10.18653/v1/2020.acl-main.104.
- Zhu, H., He, C., Fang, Y., Ge, B., Xing, M., Xiao, W., 2020. Patent automatic classification based on symmetric hierarchical convolution neural network. *Symmetry* 12. URL: <https://www.doi.org/10.3390/sym12020186>, doi:10.3390/sym12020186.

Matteo Marcuzzo is a Ph.D. student in Computer Science at the Ca' Foscari University of Venice. He received a BSc in Computer Games Technology from the University of the West of Scotland in Paisley, UK, and a MSc in Computer Science from the University of Padua, Italy. He is interested in Natural Language Processing, Representation learning and Interpretable AI.

Alessandro Zangari is a Ph.D. student in Computer Science at the Ca' Foscari University of Venice. He received his Master's Degree in Computer Science from the University of Padua in 2020. He is currently researching Deep Learning applications for Natural Language Processing algorithms and Recommendation Systems.

Michele Schiavinato is an associate researcher for the Ca' Foscari Department of Management working in the fields of machine learning, and computer vision. He started his academic career at the Ca' Foscari University of Venice, receiving a Ph.D. in Computer Science. He was consequently granted a postdoc fellowship to work on several applied research projects.

Matteo Rizzo is a Ph.D. student in Computer Science at Ca' Foscari University (Venice, Italy). Previously, he obtained a BSc and MSc in Computer Science at the University of Padua, Italy. Matteo's research mainly regards AI in agriculture and the explainability of deep learning models, focusing on interpretability and explainability.

Andrea Gasparetto received an MSc degree in Computer Science from the University of Venice, Italy, in 2012 and received his Ph.D. degree in computer science from the Ca' Foscari University. Since 2016 he is a Researcher and Teaching Assistant at the Management Department of Ca' Foscari University, working in the computer vision and shape analysis fields.

Andrea Albarelli is a researcher in the field of Artificial Intelligence, with a special focus on the design of disruptive data-driven methodologies to be applied to real-world scenarios. To this end, he works in close collaboration with companies willing to undertake a radical digital transformation process. He is currently a professor for the multidisciplinary Master Program in Data Analytics for Business and Society at the Ca' Foscari University of Venice, where he is responsible for Artificial Intelligence teaching.