Efficient Strategies for Hierarchical Text Classification: External Knowledge and Auxiliary Tasks

Kervy Rivas Rojas, Gina Bustamante, Arturo Oncevay[‡], Marco A. Sobrevilla Cabezudo[†]
Research Group on Artificial Intelligence, Pontificia Universidad Católica del Perú, Peru
[‡]School of Informatics, University of Edinburgh, Scotland
[†]Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, Brazil
`k.rivas@pucp.pe, gina.bustamante@pucp.edu.pe,`
`a.oncevay@ed.ac.uk, msobrevillac@usp.br`

Abstract

In hierarchical text classification, we perform a sequence of inference steps to predict the category of a document from top to bottom of a given class taxonomy. Most of the studies have focused on developing novels neural network architectures to deal with the hierarchical structure, but we prefer to look for efficient ways to strengthen a baseline model. We first define the task as a sequence-to-sequence problem. Afterwards, we propose an auxiliary synthetic task of bottom-up-classification. Then, from external dictionaries, we retrieve textual definitions for the classes of all the hierarchy's layers, and map them into the word vector space. We use the class-definition embeddings as an additional input to condition the prediction of the next layer and in an adapted beam search. Whereas the modified search did not provide large gains, the combination of the auxiliary task and the additional input of class-definitions significantly enhance the classification accuracy. With our efficient approaches, we outperform previous studies, using a drastically reduced number of parameters, in two well-known English datasets.

1 Introduction

Hierarchical text classification (HTC) aims to categorise a textual description within a set of labels that are organized in a structured class hierarchy (Silla and Freitas, 2011). The task is perceived as a more challenging problem than flat text classification, since we need to consider the relationships of the nodes from different levels in the class taxonomy (Liu et al., 2019).

Both flat text classification and HTC have been tackled using traditional machine learning classifiers (Liu et al., 2005; Kim et al., 2006) or deep neural networks (Peng et al., 2018; Conneau et al., 2017). Nevertheless, the majority of the latest approaches consider models with a large number of parameters that require extended training time. In the flat-classification scenario, some studies have addressed the problem of efficiency by proposing methods that do not focus on the model architecture, but in external ways of improving the results (Joulin et al., 2017; Howard and Ruder, 2018). However, the listed strategies are still underdeveloped for HTC, and the most recent and effective methods are still computationally expensive (Yang et al., 2019; Banerjee et al., 2019).

The described context opens our research question: How can we improve HTC at a lower computational cost? Therefore, our focus and main contributions are:

- A robust model for HTC, with few parameters and short training time, that follows the paradigm of sequence-to-sequence learning.

- The practical application of an auxiliary (and not expensive) task that strengthens the model capacity for prediction in a bottom-up scheme.

- An exploration of strategies that take advantage of external information about textual definition of the classes. We encode the definitions in the word vector space and use them in: (1) each prediction step and (2) an adapted beam search.

2 Efficient strategies for hierarchical text classification

2.1 Sequence-to-sequence approach

Hierarchical classification resembles a multi-label classification where there are hierarchical relationships between labels, i. e., labels at lower levels are conditioned by labels at higher levels in the hierarchy. For that reason, we differ from previous work and address the task as a sequence-to-sequence problem, where the encoder receives a textual description and the decoder generates a class at each

step (from the highest to the lowest layer in the hierarchy). Our baseline model thereafter is a sequence-to-sequence neural network (Sutskever et al., 2014) composed of:

Embedding layer: To transform a word into a vector w_i, where $i \in \{1,...,N\}$ and N is the number of tokens in the input document. We use pre-trained word embeddings from Common Crawl (Grave et al., 2018) for the weights of this layer, and we do not fine-tune them during training time.

Encoder: It is a bidirectional GRU (Cho et al., 2014) unit that takes as input a sequence of word vectors and computes a hidden vector h_i per each i time step of the sequence.

Attention layer: We employ the attention variant of Bahdanau et al. (2015), and generate a context vector a_i for each encoder output h_i.

Decoder: To use the context a_i and hidden h_i vectors to predict the $c_{l_{jk}}^{l_j}$ class of the hierarchy, where $j \in \{1,...,M\}$. M is the number of levels in the class taxonomy, l_j represents the j-th layer of the hierarchy, and l_{jk} is the k-th class in level l_j. Similar to the encoder, we use a bidirectional GRU.

2.2 Auxiliary task

For an input sequence of words, the model predicts a sequence of classes. Given the nature of recurrent neural networks, iterating over a sequence stores historical information. Therefore, for the last output computation we could take the previous inputs into consideration.

Previous work in HTC (Kowsari et al., 2017; Sinha et al., 2018) usually starts by predicting the most general category (Parent node) and continues to a more specific class (Child nodes) each time. However, by following the common approach, the prediction of the most specific classes will have a smaller impact than the more general ones when the error propagates. In this way, it could be harder to learn the relationship of the last target class with the upper ones.

Inspired by reversing the order of words in the input sequence (Sutskever et al., 2014), we propose an auxiliary synthetic task that changes the order of the target class levels in the output sequence. In other words, we go upward from the child nodes to the parent. With the proposed task, the parent and child nodes will have a similar impact on the error

propagation, and the network could learn more robust representations.

2.3 Class-definition embeddings for external knowledge integration

We analyze the potential of using textual definitions of classes for external knowledge integration. For each class $c_{l_{jk}}^{l_j}$ in any level l_j of the hierarchy, we could obtain a raw text definition from an external dictionary to compute a vector representation cv, that from now on we call the class definition vector (CDV). We thereafter use the CDV representations with the two following strategies.

2.3.1 Parent node conditioning (PNC)

For a given document D, we classify it among the target classes $C = (c_{l_{1k}}^{l_1},...,c_{l_{Mk}}^{l_M})$, where M is the number of layers in the taxonomy. In our approach, we predict the highest-level class $c_{l_{1k}}^{l_1}$ and then use its CDV representation $cv_{l_{1k}}^{l_1}$ as an additional input (alongside the encoder outputs) to the attention layer for the prediction of the next level class $c_{l_{2k}}^{l_2}$. We continue the process for all the layers of the class hierarchy.

2.3.2 Adapted beam search

Beam search is a search strategy commonly used in neural machine translation (Freitag and Al-Onaizan, 2017), but the algorithm can be used in any problem that involves word-by-word decoding. We assess the impact of applying beam search in HTC, and introduce an adapted version that takes advantage of the computed CDV representations:

$$\sum_{i=0}^{T} log P(y^i | x, y^1, ..., y^{t-1}) + CD(z, y^i) \quad (1)$$

In each step of the decoding phase, we predict a class that belongs to the corresponding level of the class hierarchy. Given a time step i, the beam search expands all the k (beam size) possible class candidates and sort them by their logarithmic probability. In addition to the original calculation, we compute the cosine distance between the CDV of a class candidate and the average vector of the word embeddings from the textual description z that we want to classify (CD component in Equation 1). We add the new term to the logarithmic probability of each class candidate, re-order them based on the new score, and preserve the top-k candidates.

Our intuition behind the added component is similar to the shallow fusion in the decoder of a

	WOS	DBpedia
Number of documents	46,985	342,782
Classes in level 1	7	9
Classes in level 2	143	70
Classes in level 3	NA	219

Table 1: Information of WOS and DBPedia corpora

neural machine translation system (Gulcehre et al., 2017). Thus, the class-definition representation might introduce a bias in the decoding, and help to identify classes with similar scores in the classification model.

3 Experimental setup

Datasets. We test our model and proposed strategies in two well-known hierarchical text classification datasets previously used in the evaluation of state-of-the-art methods for English: Web of Science (WOS; Kowsari et al., 2017) and DBpedia (Sinha et al., 2018). The former includes parent classes of scientific areas such as Biochemistry or Psychology, whereas the latter considers more general topics like Sports Season, Event or Work. General information for both datasets is presented in Table 1.

Model, hyper-parameters and training. We use the AllenNLP framework (Gardner et al., 2018) to implement our methods. Our baseline consists of the model specified in §2.1. For all experiments, we use 300 units in the hidden layer, 300 for embedding size, and a batch size of 100. During training time, we employ Adam optimiser (Kingma and Ba, 2014) with default parameters ($\beta_1 = 0.9, \beta_2 = 0.98, \varepsilon = 10^{-9}$). We also use a learning rate of 0.001, that is divided by ten after four consecutive epochs without improvements in the validation split. Furthermore, we apply a dropout of 0.3 in the bidirectional GRU encoder-decoder, clip the gradient with 0.5, and train the model for 30 epochs. For evaluation, we select the best model in the validation set of the 30 epochs concerning the accuracy metric.

Settings for the proposed strategies.

- For learning with the auxiliary task, we interleave the loss function between the main prediction task and the auxiliary task (§2.2) every two epochs with the same learning rate. We aim for both tasks to have equivalent relevance in the network training.

- To compute the class-definition vectors, we extract the textual definitions using the Oxford Dictionaries API[1]. We vectorize each token of the descriptions using pre-trained Common Crawl embeddings (the same as in the embedding layer) and average them.

- For the beam search experiments, we employ a beam size *(k)* of five, and assess both the original and adapted strategies. We note that the sequence-to-sequence baseline model use a beam size of one[2].

4 Results and discussion

Table 2 presents the average accuracy results of our experiments with each proposed method over the test set. For all cases, we maintain the same architecture and hyper-parameters in order to estimate the impact of the auxiliary task, parent node conditioning, and the beam search variants independently. Moreover, we examine the performance of the combination of our approaches[3].

In the individual analysis, we observe that the parent node conditioning and the auxiliary task provides significant gains over the seq2seq baseline, which support our initial hypothesis about the relevance of the auxiliary loss and the information of the parent class. Conversely, we note that the modified beam search strategy has the lowest gain of all the experiments in WOS, although it provides one of the best scores for DBpedia. One potential reason is the new added term for the k-top candidates selection (see Eq. 1), as it strongly depends on the quality of the sentence representation. The classes of WOS includes scientific areas that are usually more complex to define than the categories of the DBpedia database[4].

We also notice that the accuracy increment is relatively higher for all experiments on the WOS corpus than on DBpedia. A primary reason might be the number of documents in each dataset, as DBpedia contains almost seven times the number

---

[1] https://developer.oxforddictionaries.com/

[2] In preliminary experiments, we considered a beam size of ten, but we did not note a significant improvement.

[3] We tried all the possible combinations, but only report the ones that offer an improvement over the individual counterparts.

[4] Averaging words vectors to generate a sentence embedding is an elemental approach. Further work could explore the encoding of the class-definition embeddings directly from the training data, or to weight the scores of the classification model and the similarity score to balance the contribution of each term.

		WOS	DBpedia
Individual strategies	seq2seq baseline	78.84 ± 0.17	95.12 ± 0.01
	Auxiliary task	$^*78.93 \pm 0.52$	$^*95.21 \pm 0.16$
	Parent node conditioning (PNC)	$^*79.01 \pm 0.18$	$^*95.26 \pm 0.09$
	Beam search (original)	$^*78.90 \pm 0.25$	$^*95.25 \pm 0.01$
	Beam search (modified)	$^*78.90 \pm 0.28$	$^*95.26 \pm 0.01$
Combined strategies	Auxiliary task + PNC [7M params.]	$^*79.79 \pm 0.45$	$^*95.23 \pm 0.13$
	Beam search (original) + PNC	$^*79.18 \pm 0.19$	$^*\mathbf{95.30} \pm 0.10$
	Beam search (modified) + PNC	$^*79.18 \pm 0.23$	$^*95.30 \pm 0.11$
	Auxiliary task + PNC + Beam search (orig.)	$^*\mathbf{79.92} \pm 0.51$	$^*95.26 \pm 0.12$
	Auxiliary task + PNC + Beam search (mod.)	$^*79.87 \pm 0.49$	$^*95.26 \pm 0.12$
Previous work	HDLTex (Kowsari et al., 2017) [5B params.]	76.58	92.10
	Sinha et al. (2018) [34M params.]	77.46	93.72

Table 2: Test accuracy (↑ higher is better) for our proposed strategies, tested separately and combined, and a comparison with previous classifiers. Reported values are averaged across five runs, and * indicates Almost Stochastic Dominance (Dror et al., 2019) over the seq2seq baseline with a significance level of 0.05. The amount of parameters of each combined strategies is up to seven million.

of documents of WOS. If we have a large number of training samples, the architecture is capable of learning how to discriminate correctly between classes only with the original training data. However, in less-resourced scenarios, our proposed approaches with external knowledge integration could achieve a high positive impact.

As our strategies are orthogonal and focus on different parts of the model architecture, we proceed to combine them and assess their joint performance. In the case of WOS, we observe that every combination of strategies improves the single counterparts, and the best accuracy is achieved by the merge of the auxiliary task and PNC, but with an original beam search of size five. Concerning DBpedia, most of the results are very close to each other, given the high accuracy provided since the seq2seq baseline. However, we note the relevance of combining the PNC strategy with the original or modified beam search to increase the performance.

Finally, we compare our strategies to the best HTC models reported in previous studies (Kowsari et al., 2017; Sinha et al., 2018). We then observe that the results of our methods are outstanding in terms of accuracy and number of parameters. Moreover, the training time of each model takes around one hour (for the 30 epochs), and the proposed auxiliary task do not add any significant delay.

5 Related work

Most of the studies for flat text classification primarily focus on proposing a variety of novel neural architectures (Conneau et al., 2017; Zhang et al., 2015). Other approaches involve a transfer learning step to take advantage of unlabelled data. McCann et al. (2017) used the encoder unit of a neural machine translation model to provide context for other natural language processing models, while Howard and Ruder (2018) pre-trained a language model on a general-domain monolingual corpus and then fine-tuned it for text classification tasks.

In HTC, there are local or global strategies (Silla and Freitas, 2011). The former exploits local information per layer of the taxonomy, whereas the latter addresses the task with a single model for all the classes and levels. Neural models show excellent performance for both approaches (Kowsari et al., 2017; Sinha et al., 2018). Furthermore, other studies focus on using transfer learning for introducing dependencies between parent and child categories (Banerjee et al., 2019) and deep reinforcement learning to consider hierarchy information during inference (Mao et al., 2019).

The incorporation of external information in neural models has offered potential in different tasks, such as in flat text classification. By using categorical metadata of the target classes (Kim et al., 2019) and linguistic features at word-level (Margatina et al., 2019), previous studies have notably improved flat-text classification at a moderate com-

putational cost. Besides, Liu et al. (2016) outperform several state-of-the-art classification baselines by employing multitask learning.

To our knowledge, the latter strategies are not explicitly exploited for HTC. For this reason, our study focuses on the exploration and evaluation of methods that enable hierarchical classifiers to achieve an overall accuracy improvement with the least increasing complexity as possible.

6 Conclusion

We presented a bag of tricks to efficiently improve hierarchical text classification by adding an auxiliary task of reverse hierarchy prediction and integrating external knowledge (vectorized textual definitions of classes in a parent node conditioning scheme and in the beam search). Our proposed methods established new state-of-the-art results with class hierarchies on the WOS and DBpedia datasets in English. Finally, we also open a path to study integration of knowledge into the decoding phase, which can benefit other tasks such as neural machine translation.

Acknowledgements

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Siddhartha Banerjee, Cem Akkaya, Francisco Perez-Sorrosal, and Kostas Tsioutsiouliklis. 2019. Hierarchical transfer learning for multi-label text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6295–6300, Florence, Italy. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. 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*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1107–1116, Valencia, Spain. Association for Computational Linguistics.

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance - how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785, Florence, Italy. Association for Computational Linguistics.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60, Vancouver. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, and Yoshua Bengio. 2017. On integrating a language model into neural machine translation. *Computer Speech & Language*, 45:137 – 148.

Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *56th Annual Meeting of the Association for Computational Linguistics*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Jihyeok Kim, Reinald Kim Amplayo, Kyungjae Lee, Sua Sung, Minji Seo, and Seung-won Hwang. 2019. Categorical metadata representation for customized

text classification. *Transactions of the Association for Computational Linguistics*, 7:201–215.

Sang-Bum Kim, Kyoung-Soo Han, Hae-Chang Rim, and Sung Hyon Myaeng. 2006. Some effective techniques for naive bayes text classification. *IEEE Trans. on Knowl. and Data Eng.*, 18(11):1457–1466.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. HDLTex: Hierarchical deep learning for text classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 364–371. IEEE.

Liqun Liu, Funan Mu, Pengyu Li, Xin Mu, Jing Tang, Xingsheng Ai, Ran Fu, Lifeng Wang, and Xing Zhou. 2019. NeuralClassifier: An open-source neural hierarchical multi-label text classification toolkit. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–92, Florence, Italy. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 2873–2879. AAAI Press.

Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. 2005. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36–43.

Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. 2019. Hierarchical text classification with reinforced label assignment. 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)*, pages 445–455, Hong Kong, China. Association for Computational Linguistics.

Katerina Margatina, Christos Baziotis, and Alexandros Potamianos. 2019. Attention-based conditioning methods for external knowledge integration. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3944–3951, Florence, Italy. Association for Computational Linguistics.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 1063–1072, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Carlos N. Silla and Alex A. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Koustuv Sinha, Yue Dong, Jackie Chi Kit Cheung, and Derek Ruths. 2018. A hierarchical neural attention-based text classifier. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 817–823, Brussels, Belgium. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Pengcheng Yang, Fuli Luo, Shuming Ma, Junyang Lin, and Xu Sun. 2019. A deep reinforced sequence-to-set model for multi-label classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5252–5258, Florence, Italy. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.