Hierarchical classification of educational resources

Gregor Žunič
Jožef Stefan Institute
Ljubljana, Slovenia
gregor.zunic@ijs.si

Erik Novak
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia
erik.novak@ijs.si

ABSTRACT

This paper describes an approach to automate the process of labelling hierarchically structured data. We propose a top-down level-based approach with SVMs to classify the data with scientific domain labels. The model was trained on labeled open education lectures and returns high accuracy predictions for lectures in the English language. We found that our model performs better with the traditional text extraction method TF-IDF than with pre-trained language model XLM-RoBERTa.

KEYWORDS

hierarchical classification, support vector machine, multi-class classification, machine learning, open educational resources

ACM Reference Format:
Gregor Žunič and Erik Novak. 2020. Hierarchical classification of educational resources. In *Proceedings of Slovenian KDD Conference (SiKDD'20).* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Manually labeling data can be tedious work; one must have sufficient background knowledge about the data and have clear instructions in the labeling process. This becomes even more difficult when the data needs to be annotated with hierarchically structured labels.

In this paper we present a top-down level-based approach using support vector machines (SVMs) for labeling open education resources (OERs). The labels are in a hierarchical structure and represent different scientific domains. We compare different lecture representations using TF-IDF and XLM-RoBERTa and find that the TF-IDF representations yield better results. Even though the paper focuses on OERs the method can be generalized to any textual data set.

The remainder of the paper is structured as follows. Section 2 describes the related work done on the topic of hierarchical classification. Next, we present the data used in the evaluation in Section 3. The methodology is described in Section 4. The evaluation setting and its results are described in Section 5 followed by a discussion in Section 6. We present the future work in Section 7 and conclude the paper in Section 8.

SiKDD'20, October 2020, Ljubljana, Slovenia

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

2 RELATED WORK

There are two approaches to hierarchically classify the data: (1) the Big-bang, and (2) the Top-down level-based approach [4, 8, 9].

The big-bang approach works by training (complex) global classifiers which consider the entire class hierarchy as a whole. Each global classifier is binary and decides if the material fits the entire hierarchy (entire hierarchy is for example "Computer Science/Machine Learning/Support Vector Machine"). The advantage of this approach is that it avoids class-prediction inconsistencies across multiple levels. The major drawback of this approach is the high complexity due to the enforcing the model to correctly predict the whole hierarchy branch, which can be difficult to achieve.

The top-down level-based approach works by training local classifiers at each level to distinguish between its child nodes. An example will first, at the root level, be classified into a second-level category. It will then be further classified at the lower level category until it reaches one or more final categories where it can not be classified any further. The main advantage of this model is its simplicity. The disadvantage is the difficulty to detect an error in the parent category which could lead to false classification.

The most common implementation of a local classifier [3] is the support vector machine [7, 11]. In the later papers they propose to train separate SVMs for every level of a branch in the hierarchy.

3 DATA SET

The data set used in the experiment consists of 28,769 OER lectures available at Videolectures.NET [10], an award winning video OER repository. For each lecture we collected the following metadata: title, description, labels, language, authors, date published and the length of the lecture. The description is present in 58% of the lectures. The data set contains 24532 lectures in English, 3930 in Slovene and 307 lectures in other 16 languages.

Preprocessing. For our methodology we used only the lecture's title, description, language and categories. Each lecture is labeled with one or more scientific (sub-)domains most relevant for the lecture (e.g. "Computer Science", "Computer Science/Crowd Sourcing"). Figure 1 shows the distribution of lectures per number of labels.

Almost half of the lectures have more than one label. Lectures with no labels are placed under the "No Labels" category. These lectures are mostly introductory speakers' presentations in conferences. We focus on predicting a single label with high accuracy. We prescribed to only have one label per lecture. We achieve this by duplicating a lecture n times, where n is the number of labels of the lecture and assign a distinct label to each duplicate. Although the duplicates may reduce the performance of the models we do not reduce the already small number of lectures used during the

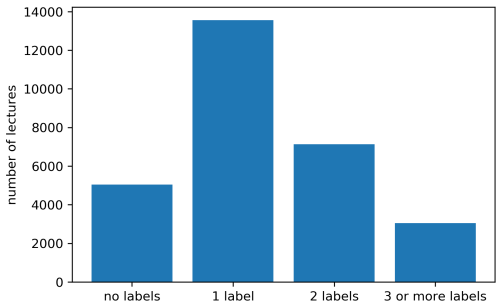

Figure 1: Distribution of lectures per number of corresponding labels. Most of the lectures have only one label.

training process. Figure 2 shows the top scientific domain labels in the data set.

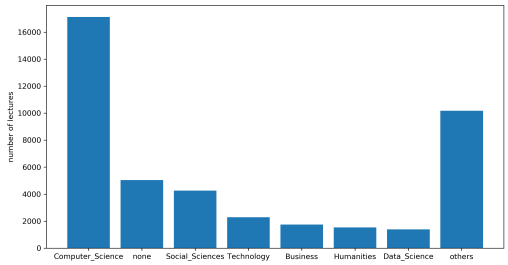

Figure 2: Top scientific domain labels in the data set. The most frequent label is Computer_Science.

The most frequent label is "Computer Science". In addition, a large number of lectures are not labeled; this is because a lot of lectures are presentations that do not correspond to any of the scientific domains. The data set is unbalanced on both domain and sub-domain levels.

4 METHODOLOGIES

In this section we describe the methods used to perform the feature extraction of the text, the implementation of multi class classifier model and the lectures' weights.

The input to the classifier is a raw string created by concatenating the title and the description if the description is available. It is then converted to a vector. In this paper we experimented with two approaches: TF-IDF and XLM-RoBERTa.

4.1 Feature Extraction

TF-IDF. Each lecture is represented with a vector of its TF-IDF values [6]. TF measures how frequently a term occurs in a lecture's text. The IDF is a measure of how much information the word provides. If it is common across all lectures its value is close to 0. The terms with the highest TF-IDF scores are usually the ones that characterize the topic of the lecture best.

The size of the lecture's vector representation is exactly the same as the total number of unique words. Since most of the features are zero the lecture vectors are sparse.

XLM-RoBERTa. The model is based on the RoBERTa model released in 2019. It is a large language model trained on 2.5 TB of CommonCrawl data [2]. The model achieves state-of-the-art performance on cross-lingual classification, sequence labeling and question answering. The most useful feature of the model is that it does not require the sentence language as an input. In theory, it extracts the same vectors for similar words in 100 languages.

The length of the vector that the model outputs is 768. To extract the features a CUDA-enabled GPU is required and the model training is very slow.

4.2 Multi-class SVM Classifier

We chose the top-down level-based approach for our classifier. The raw text input is firstly vectorized following one of the two feature extraction approaches described in Section 4.1. The vector is then input to the main SVM which determines the first category. Then the input is handled by the second SVM, trained specifically for sub-labels of first classified category. If a sub-label tops the threshold of 0, this step is repeated, otherwise the model outputs the lowest level parent category.

For example "Computer Science" is the first determined category. Then the input is handled by the SVM trained on sub-labels of "Computer Science", which determines that the input does not match with any of the sub-labels. The model puts the lecture in the "Computer Science" category. This is visually explained in figure 3.

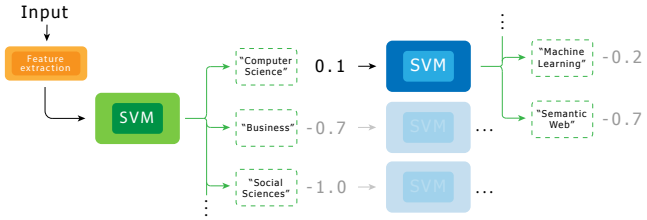

Figure 3: Visual representation of hierarchical SVM classifier. The example shows a lecture classified as belonging to the "Computer Science" category

Each SVM is an implementation of a multi-class classifier using the one-vs-rest approach. Predicted class should always be dominant otherwise the recommendation is not relevant.

4.3 Lecture Weights

Each lecture is assigned a weight of $\frac{1}{n^x}$, $x = 4$, where n is the number of total labels in the original lecture and x is a parameter. If $x < 4$ the accuracy is greatly reduced, if $x > 4$ the accuracy is increased by a small margin. It converges when $x \to \infty$. When increasing the parameter x the weight comes closer to 0 which means that the model accounts for data less during training. This means that the 4^{th} power is a sufficient balance between excluding some data and reducing the accuracy.

The other approach could be to ignore multi-label lectures during testing phase ($\frac{1}{n^\infty}$).

Because some labels are so scarce, we limit ourselves to labels with at least 20 lectures. This reduces the total number of labels in the data set from 502 to 244.

5 EVALUATION

5.1 Parameters and Specifications

SVM. The SVM implementation used in the evaluation is the LinearSVC [1] with the default parameters.

XLM-RoBERTa. The model used for representation generation is the hugging face's pretrained model [5] which was trained on default parameters found in the paper [2]. The training was executed on the Google Colab (online hosted Jupyter notebook) free tier machine (12GB RAM, dual core CPU, NVIDIA K80).

5.2 Results

Table 1 shows the performance of the different models with linear kernel. We have also evaluated other kernels (polynomial, RBF, sigmoid), but the performance was worse than using linear kernel. That is why we omitted them from the performance table.

TF-IDF with linear kernel SVM. Using the TF-IDF method for feature extraction we found that the SVMs performed the best with linear kernel. One explanation for such results is that the dimension of the features is large (more than 60k), which means that other more advance kernels might lead to over-fitting.

XLM-RoBERTa with linear kernel SVM. The model's performance was worse than using TF-IDF. The accuracy of the main classifier was 19% compared to 70% when using TF-IDF. The other SVM kernels (polynomial, RBF, sigmoid) performed worse compared to linear kernel. Table 1 shows the performance of the model.

SVM. The problem with current SVM implementation is that it can only put the lecture in one category. One way to solve the issue of only one label would be to firstly predict one label. Then, if the user (editor) wants another prediction, the model can output the prediction with second highest certainty.

TF-IDF vs XLM-RoBERTa. The advantage of choosing XLM-RoBERTa over of TF-IDF is that it works with 100 languages. The vector outputs are similar [2] for all languages. This was proven by translating the same text input into multiple languages (using Google Translate) and the predicted category did not change. When using TF-IDF you have to split the original data set into subsets containing a single language and train the model from scratch. That would be possible with enough data. For some languages (German, French) the the data set contains less than 30 lectures, which means that you can not train an SVM sufficiently.

6 DISCUSSION

Unbalanced Data Set. We found the SVM trained on an oversampled data set to be working better than the SVM trained on the raw data set. Due to the unbalanced data if the data set is not resampled the bias towards the strongest category (*Computer Science*) is strongly presented. For example neutral words such as " ", "the" etc. are classified as belonging in *Computer Science* category.

Comparing Word Embedding Techniques. The TF-IDF approach performs much better than XLM-RoBERTa which is surprising. Pre-trained models usually perform better than legacy feature extractors. The reason could be that the hyper parameters of the model were not set correctly, but we did not find the right balance for the model to perform any better. The production versions could include both models. For languages with a lot of data in the data set, the model would opt for SVMs trained on features extracted using TF-IDF, because of the better performance. All other languages would be handled by SVMs trained by XLM-RoBERTa, because the classifier performs much better than random.

The TD-IDF method could also be used to classify lectures that are in the non-english languages by firstly translating the text to English before using them during training. With this approach the model could work in all languages and retain the simplicity of TF-IDF. Note that that this approach would be strongly dependant on the quality of the translations.

Weighting the errors during the training process. We did not use the hierarchy structure for calculating the error between the predicted and the actual labels hence all the errors types during training were the same. This is not ideal because the error should be more significant when the classifier incorrectly predicts the main branch versus when it incorrectly predicts a lower level label. For example, if we take a lecture that is labeled as "Computer Science/Machine Learning" then the error should be bigger if our classifier predicts the "Biology" label rather than the "Computer Science/Semantic Web" label.

7 FUTURE WORK

We intend to improve the performance of the XLM-RoBERTa and to experiment with other language models and try to achieve better performance.

One additional direction for future work might be training a multiclass classifier to predict more than one label to a given lecture. We tried implementing the multi label output classifier using the MultiOutputClassifier wrapper on SVM but the precision of the model was noticeably lower.

The model is ready to be used in production in Videolectures.NET as a recommender engine to help the editors. The service could either be wrapped in a *Flask* microservice or directly into Videolectures.NET's backend.

8 CONCLUSION

In this paper we explore a top-down level-based approach for classifying OER lectures with scientific domain labels. We used oversampling to handle label unbalance and experimented with two text representation approaches, TF-IDF and XLM-RoBERTa. We found that the model using the TF-IDF representations gives better results.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and X5GON European Unions Horizon 2020 project under grant agreement No 761758.

REFERENCES

[1] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 108–122.

[2] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. *arXiv preprint arXiv:1911.02116* (2019).

parent category	TF-IDF acc.	TF-IDF recc.	TF-IDF F	TF-IDF prec.	XLM-RoBERTa acc.	XLM-RoBERTa recc.	XLM-RoBERTa F	XLM-RoBERTa prec.	materials
Root	70%	69%	72%	75%	19%	11%	19%	68%	27009
Computer Science	59%	59%	60%	61%	9%	4%	8%	50%	12935
Machine Learning	60%	55%	59%	64%	11%	5%	9%	26%	3260
Semantic Web	75%	71%	75%	79%	23%	20%	31%	68%	454
Computer Vision	82%	79%	81%	83%	57%	55%	59%	63%	140
Social Sciences	73%	72%	73%	74%	35%	24%	34%	60%	2928
Society	74%	72%	72%	72%	36%	28%	38%	60%	890
Politics	76%	66%	75%	86%	59%	43%	54%	73%	83
Law	96%	96%	96%	96%	57%	41%	51%	67%	112
Journalism	100%	100%	100%	100%	91%	88%	90%	92%	53
Technology	84%	82%	82%	82%	50%	43%	50%	60%	970
Nanotechnology	69%	59%	69%	83%	46%	37%	46%	62%	78
Business	74%	72%	73%	74%	43%	36%	43%	54%	1009
Transportation	63%	53%	61%	71%	33%	22%	32%	56%	267
Humanities	85%	83%	84%	85%	55%	48%	55%	65%	873
Biology	71%	66%	67%	68%	23%	17%	22%	31%	430
Science	78%	77%	78%	79%	53%	51%	52%	53%	656
Medicine	89%	88%	89%	90%	39%	34%	48%	83%	326
Computers	83%	83%	83%	83%	55%	48%	53%	59%	731
Mathematics	89%	87%	89%	91%	41%	36%	38%	40%	421
Physics	86%	81%	85%	89%	36%	32%	38%	46%	227
Arts	88%	87%	85%	83%	45%	40%	49%	63%	338
Visual Arts	100%	100%	100%	100%	62%	56%	70%	92%	159
Design	52%	46%	55%	68%	23%	9%	14%	30%	104
Chemistry	100%	100%	100%	100%	85%	83%	91%	100%	161
Environment	94%	94%	93%	92%	71%	66%	73%	81%	161
Earth Sciences	73%	67%	74%	82%	50%	51%	50%	49%	27

Table 1: Comparison of model performance using the linear kernel. The performance of the TF-IDF approach is better than that of XLM-RoBERTa.

[3] Susan Dumais and Hao Chen. 2000. Hierarchical Classification of Web Content. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '00)*. Association for Computing Machinery, New York, NY, USA, 256–263. https://doi.org/10.1145/345508.345593

[4] A. D. Gordon. 1987. A Review of Hierarchical Classification. *Journal of the Royal Statistical Society: Series A (General)* 150, 2 (1987), 119–137. https://doi.org/10.2307/2981629 arXiv:https://rss.onlinelibrary.wiley.com/doi/pdf/10.2307/2981629

[5] huggingface. 2020. huggingface.co - pretrained models. https://huggingface.co/transformers/pretrained_models.html.

[6] J.D. Rajaraman, A.; Ullman. 2011. Mining of Massive Datasets. pp. 1–17. http://i.stanford.edu/~ullman/mmds/ch1.pdf.

[7] Ahmad Shalbaf, Reza Shalbaf, Mohsen Saffar, and Jamie Sleigh. 2020. Monitoring the level of hypnosis using a hierarchical SVM system. *Journal of Clinical Monitoring and Computing* 34, 2 (2020), 331–338. https://doi.org/10.1007/s10877-019-00311-1

[8] Carlos N. Silla and Alex A. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22, 1 (2011), 31–72. https://doi.org/10.1007/s10618-010-0175-9

[9] Aixin Sun, Ee-Peng Lim, and Wee-Keong Ng. 2003. Performance measurement framework for hierarchical text classification. *Journal of the American Society for Information Science and Technology* 54, 11 (2003), 1014–1028. https://doi.org/10.1002/asi.10298 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.10298

[10] VideoLectures.Net. 2020. VideoLectures.NET - VideoLectures.NET. https://videolectures.net/. Accessed: 2020-08-20.

[11] S. V. M. Vishwanathan and M. Narasimha Murty. 2002. SSVM: a simple SVM algorithm. 3 (2002), 2393–2398 vol.3.