Proceedings of the 11^{th} International Conference on Applied Informatics
Eger, Hungary, January 29–31, 2020, published at http://ceur-ws.org

Automatic Tag Recommendation for News Articles

Zijian Győző Yang[abc], **Attila Novák**[bc], **László János Laki**[bc]

[a]Eszterházy Károly University, Faculty of Informatics, Hungary
yang.zijian.gyozo@uni-eszterhazy.hu

[b]MTA-PPKE Hungarian Language Technology Research Group, Hungary

[c]Pázmány Péter Catholic University, Faculty of Information Technology and Bionics, Hungary
{yang.zijian.gyozo,novak.attila,laki.laszlo}@itk.ppke.hu

Abstract

In this paper, we present an automatic neural tag recommendation system for Hungarian news articles and the results of our experiments concerning the effect of preprocessing applied to the texts and various parameter settings. A novelty of the approach is a combination of subword tokenization with character-n-gram-based representations, which resulted in high gains in recall. The best system yields 76% precision at 58% recall. Subjective performance is higher, because suggested labels missing from the reference often fit the document well or are similar to missing reference labels. We also created an online GUI for the tag recommendation system that makes it possible for the user to interactively set threshold parameters facilitating customization of precision and recall.

Keywords: tag recommendation, tag suggestion, keyword generation, fastText, SentencePiece tokenizer

1. Introduction

Content published at online news sites is often labeled using thematic tags. The presence of these labels makes it possible for readers to focus on topics interesting for them and for the publisher to display content related to each individual article.

The publisher can also customize/filter/recommend content probably interesting for registered users that have a user profile. Content keywords added to the meta tags in the HTML head section also play a role in ranking algorithms of search engines. As long as the keywords are really appropriate to the content, they may positively affect search engine ranking.

Tag recommendation algorithms have existed for several years. For an overview of methods and solutions see e.g. [1] and [4]. Nevertheless, thematic tags are assigned to content in a manual manner at many online publishers. Some editorial boards include a dedicated staff member (usually an educated librarian) whose task is to assign thematic tags to all published articles, while other publishers make it the responsibility of each author to assign keywords. The former approach has a rather limited throughput. The latter approach makes the process cheaper and much more productive, however, it results in a proliferation of keywords and a much less uniform and well-thought-out keyword usage. Uniformity of keyword usage is only partially assured by tagging guidelines and prefix-based predictive input returning formerly used keywords integrated in the content management system (CMS) used by the publisher. The latter, however also results in duplication of typos in the keywords and thus may increase rather than decrease variation in many cases.

In this paper, we present an automatic keyword recommendation system for Hungarian that can be integrated in content management systems supporting editorial work. We train our models for keyword prediction using manually tagged past articles.

2. Related work

The most important former result published on automatic keyword assignment for Hungarian news was a system created for the [origo] news site [2]. The goal of that research was automatic tagging of past news articles that had been published by [origo] before manual tagging was introduced so as to make them available for automatic content recommendation. The system created was not integrated in the editorial system to support future work of the news editors or authors, or at least such application is not mentioned in the publications. The solution outlined in [2] is rather complex: the text underwent detailed automatic annotation including PoS tagging, lemmatization, name entity recognition and NP chunking. Identified NP's were normalized/lemmatized, and the type of the name entity was identified (person, location or organization). A rather limited amount of tagged content was available at that time, thus authors of that paper relied primarily on extracting and normalizing phrases present in the article when assigning keywords.

We, in contrast, could rely on a reasonably large amount of tagged documents. And our main goal is also different: automatic support of the tagging of future content, which will remain a manually controlled process. This makes a simpler solution possible, while manual control of precision vs. recall is a useful feature in a CMS-integrated computer-aided keyword assignment system. We based our solution on the fastText text classification library [3], which uses neural classifiers to assign thematic tags. The input to the classifiers is a semantic vector representation of the document computed as a weighted average of the vectors representing individual tokens and token n-grams. The latter, in turn, are computed as the average of the representation of character n-grams of various length making up individual tokens in the text. Classifiers calculate a probability estimate of each label being suitable to tag a document with the given semantic vector representation. Setting a threshold on the probability estimate can be used to distinguish suitable labels from unsuitable ones as well as to fine-tune the precision and the recall of the algorithm. Although more recent deep neural models surpass the performance of fastText for text classification (currently XLNet [9] performing best on many English text classification benchmark data sets), their complexity and resource requirements exceed that of fastText with a much higher margin than what the difference in performance would justify.

An experiment concerning the performance of fastText on Hungarian text classification was presented in [8]. However, the task in that experiment was a simple two-class (sports vs. video game) classification problem, while our goal is to select the most suitable thematic labels from a set of several thousand or tens of thousands of possible labels, where the number of suitable labels can also widely differ depending on the length and the topic of the document.

3. The tag recommendation system

In this section, we present the architecture and components of the system and the training database.

3.1. Architecture

The tag recommendation system is a REST-based web application consisting of a JavaScript-bootstrap-based front-end UI and a Python-based back-end server. The title, lead, author, date and content of the article can be entered through the front end, and it can be submitted for tagging to the back end.

Proposed keywords can be either names, thematic labels, or trend labels. Labels of the latter type pertain to unique events (a specific performance, festival, sports event, election etc.), often of periodic recurrence (such as Olympic games or elections). Trend labels are assigned to many articles published within a relatively short period of time, but then they fall out of use. However, reports on a specific fair, conference or award ceremony in an event series are quite similar to reports on any other event in that series, thus trend labels pertaining to past instances of periodically recurring events appear as noise when tagging a document on a current instance of the event. To prevent this, we distinguish trend labels from static names (of persons, organizations, products and more generic event types etc.) and from generic conceptual labels.

Suggested labels are displayed along with the confidence/probability assigned to them by the classifier, and they can be filtered by setting a threshold using a slider. In addition a minimum amount of best labels to be displayed can be set.

The front end also contains a demo where test documents taken from the original labeled data are displayed along with the original manually assigned reference keywords and those proposed by the automatic classifier. The set can be dynamically filtered using the threshold slider.

The back end is implemented in Python. It uses a Flask-based web server to communicate with the front end that uses AJAX requests to send the document to be tagged and get the suggested labels. The format of the data packages is JSON. The back end loads distinct models for static names and conceptual labels and one suggesting trend labels that is trained only on recent documents (published not earlier than 6 months before). For older documents, trend labels are substituted with their generic equivalents.

3.2. The tools

As we mentioned above, our models are based on text representation and classification models implemented in the open-source fastText library [3]. FastText was developed at Facebook and is implemented in C++. The semantic vector representations created by the neural model implemented in fastText is based on distributional properties of words. The model is trained to predict words in the context of a given word token (or vice versa). FastText handles the problem that rare words and ones unseen in the training corpus (i.e. out-of-vocabulary – OOV words) would lack a representation by inducing character n-gram, instead of word, representations. Vectors representing words and documents are calculated averaging word n-gram representations.

Recent end-to-end deep neural models for machine translation and other high-level NLP tasks use another approach, subword tokenization, to handle the serious problems former word-token-based models had: excessive memory requirements and a general inability to adequately handle rare and OOV word forms.

The subword tokenizer most frequently used in current neural machine translation systems is SentencePiece [6], a language-independent subword tokenizer and detokenizer implementing two subword tokenization algorithms, byte-pair-encoding BPE [7] and unigram language model [5]. Using a subword tokenizer makes it possible to do away with any language-specific preprocessing. It guarantees a limited vocabulary, the size of which can be specified in advance, almost entirely eliminating the problem of unknown tokens. (The only exception is possible unknown characters resulting in unknown tokens e.g. from foreign-language document sections.)

While the application of character n-grams handles the OOV problem in fastText and thus subword tokenization seems superfluous in this context, driven by a sudden impulse, we tested during our preliminary experiments whether applying subword tokenization influences the labeling performance of the classifiers. And we found that, indeed, training and testing the fastText classifiers on BPE-tokenized

		tokens				types	
corpus	articles	words	labels	names	OOV	labels	names
Weekly train	94094	46,25M	0,46M	0,44M	-	24849	23822
Weekly test	6902	3,33M	38775	37559	1036	5089	4152
Online train	328635	89,08M	1,33M	1,26M	-	186508	89711
Online test	45105	13,43M	0,21M	0,2M	14488	53568	24607

Table 1: Sizes and other features of train-test splits of both corpora

input greatly improves recall while precision is affected only to a much more limited extent. Since our goal is to use the models in a human-controlled environment where improper keywords can easily be identified and unselected by the author of the article, the slight reduction of precision is a fair price given for a largely improved recall.

3.3. The corpus

For our experiments, we used articles from the HVG printed weekly newspaper from 1994–2017 as well as news documents from the online hvg.hu news portal (2012–2018). The former were tagged by a single expert librarian, while the latter by the article authors. As a result, the set of labels used within the latter corpus is not uniform. Authors may use different (often misspelled) forms of the same label: *M0-ás autópálya, M0-ás, M0-s autópálya, M0-s, M0-ás autóút, M0, M0-s autóút* 'M0 highway, M0 motorway, M0, M0 freeway, highway M0', etc. In many cases, synonymous labels include not only different spellings, but labels of different origin or style for the same concept, for example *fű, marihuána, kannabisz* 'weed, marijuana, cannabis' etc.

Some documents in the weekly newspaper corpus are tables or graphs rather than articles. We omitted these from our experimental data.

4. Experiments and results

We performed detailed experiments on the weekly newspaper and the online news corpus concerning the effect of various factors on text classification and labeling performance. We wanted to see how the number of training examples seen for a specific label affects performance. The train-test split was done for both corpora so that we have at least 5 test documents for each label occurring at least 15 times in the corpus. Sizes and other features of the two train-test splits are shown in Table 1.

Although the weekly newspaper corpus spans 24 years while the online corpus only 7 years, the latter contains more than 3 times as many documents and, due to the much higher variation in label usage, 7.5 times more different label types. About 95% of label occurrences are names in both corpora. There is, however, a significant difference in the ratio of name labels among label types: about 96%

in the weekly corpus, while under 50% in the online corpus. This is due both to much higher variation in concept labels and sloppy lower cased spelling of many rare name labels in the online corpus. The ratio OOV label occurrences is 2.7% in the weekly corpus and 6.9% in the online corpus.

We tested different tokenization models on the weekly corpus: traditional punctuation-based tokenization, no tokenization, and BPE subword tokenization. We trained joint models where name labels were not distinguished from concept labels, and ones were they were separated. When training and predicting named entity (NE) labels separately, we preprocessed documents keeping only the maximum-two-word context of words containing capital letters. We omitted untokenized and traditionally tokenized models for the online corpus from our experiments, as we found these to have an inferior performance on the weekly corpus.

All models were trained using the same parameters. We trained one-to-many classifiers to handle variable label counts. The dimension of vectors was 100 for all models. Training models for the online corpus took much longer than for the weekly corpus due to the much higher number of different labels (much more one-to-many classifiers need to be trained). Due to this, we trained online models for only 30 epochs in contrast to 50 epochs used when training the weekly models.

We wanted to see how the model performance is affected by the frequency of labels in the training data. We thus assigned labels to bins based on their frequency, and measured precision, recall and F_1 score for labels in each bin as a function of the cutoff threshold used to select the top label candidates. We measured performance for names, non-name labels and both combined. We were especially interested in performance on rare labels. If the model cannot learn to predict rare labels, we can safely eliminate them from the training data significantly speeding up training and update of the models without affecting performance.

We present our findings in Fig. 1. The diagrams shows precision (P), recall (R) and F_1 score of each model as a function of the cutoff threshold. The diagrams on the left show performance on name labels, the ones on the right on non-name labels, the ones in the middle for all labels. We can clearly see that subword tokenization (*weekly-sp* model) results in a much higher recall and F_1 score than traditional *(weekly-tok)* or no tokenization *(weekly-untok)*. The latter two models performed almost identically. Although their precision is higher than that of the subword tokenized models at lower cutoff threshold values, they have very much lower recall. We measured lower performance on the online corpus (*online-sp* model) due to the much higher variation of labels. However, the subjective impression concerning the quality of labels suggested by this model is not worse than for models trained on the weekly corpus due to appearance of labels synonymous to the reference labels. Members of the editorial board clearly found the performance this model superior. We created a tool that can be used to merge and normalize synonymous labels. Normalization of the label set used in the online corpus is under way using this tool.

All models perform better for names than for non-name labels except that recall for non-names is higher for models trained on the weekly corpus for lower thresh-

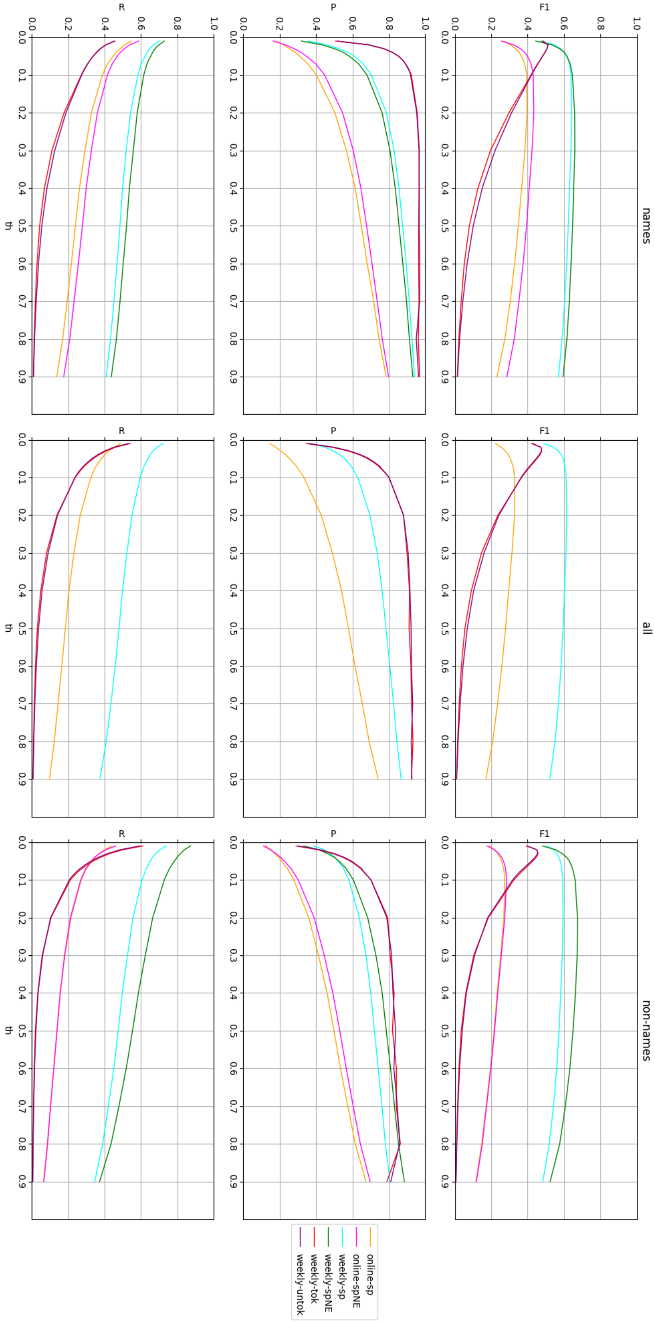

Figure 1: Performance of 6 models on the whole label set as a function of cutoff threshold: P, R, F_1 scores. Corpus: weekly/online; tokenization: untok=none, tok=punctuation, sp=BPE SentencePiece; NE: separate models for names and non-names

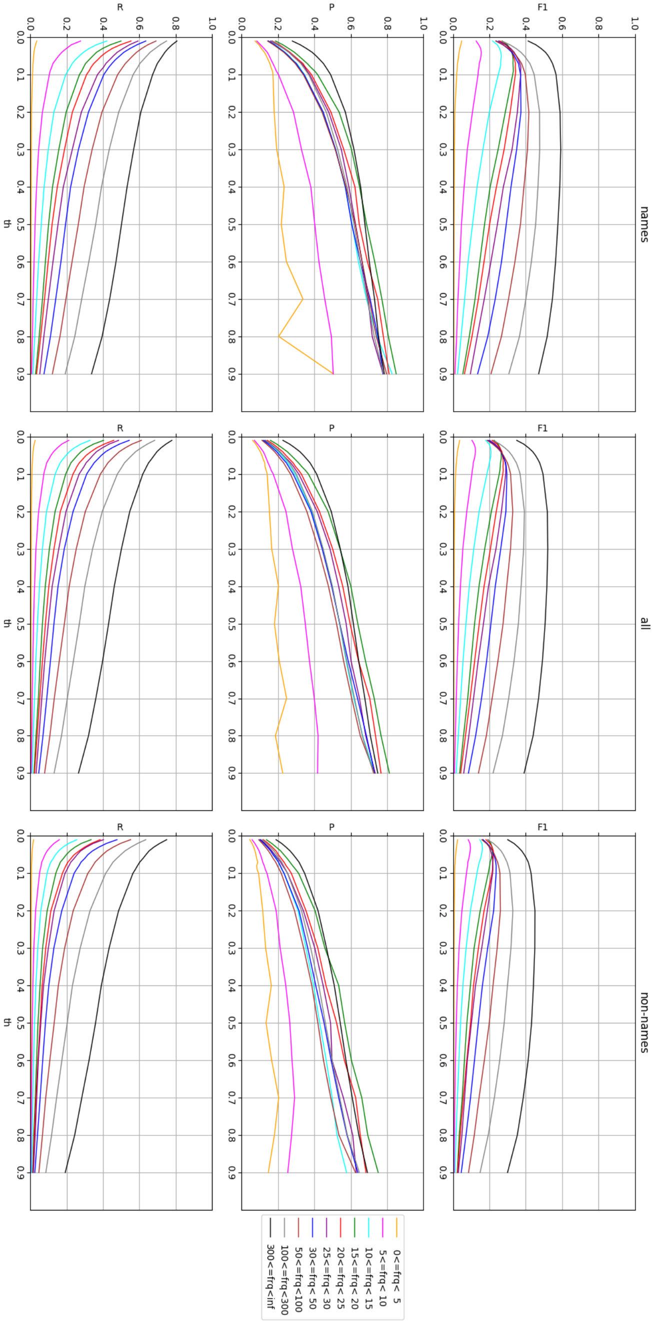

Figure 2: Online modell, BPE subword tokenizer, P, R, F_1 scores for different label frequency classes as a function of cutoff threshold

old values. For non-subword-tokenized models, there is a very pronounced peak in F_1 score at 0.02, while subword-tokenized models perform best at a threshold of 0.2, but they have a much more balanced performance overall. Training a separate model for names and non-name labels clearly improved performance with the exception of a slight drop in precision for names in the weekly corpus, but on the online corpus, name label precision also slightly improved.[1]

We also measured performance on distinct label frequency classes. For lack of space, we present only the values for the *online-sp* model in Fig. 2. All subword-tokenized models have measurable recall for all label frequency classes, although it is rather low for very rare labels (with less than 5 training examples). This means that we can safely eliminate labels of very low frequency from the training data reducing model size and training time without hurting performance. Precision is not very high either for labels with less than 10 training examples. The general trend is that the more training examples there are for a label, the higher precision and recall we get. Untokenized and traditionally tokenized models have measurable recall only at very low threshold values. While one fifth of the label occurrences in the weekly test data pertains to rare labels having less than 30 training examples, only 0.5% of these labels is actually suggested by these models above the 0.05 threshold value.

5. Conclusion

In this paper, we presented an automatic thematic keyword suggestion system for Hungarian news text. We created a web-based front end to the system that makes it possible for the user to set certain parameters, e.g. the cutoff threshold for the keyword suggestion list. This allows customization of the precision and recall of the keyword candidates. In an optimal setting, we can recommend thematic keywords with 76% precision at 58% recall. We performed a detailed evaluation of models examining various preprocessing and parameter options. Combining the fastText model with subword tokenization substantially improved recall while the decrease of precision was tolerable. At the same time, model size was also reduced to a fraction of the original. We have also found that rare labels can be eliminated from the training corpus speeding up training and reducing model size without significantly affecting performance.

Acknowledgments

This research was implemented with support provided by grant FK 125217 of the National Research, Development and Innovation Office of Hungary financed under

[1] The drop of precision on names in the weekly corpus seems mainly to be due to country labels, some of which are very frequent. Location can be inferred better relying on the whole text than just on names present in the document.

the FK 17 funding scheme as well as through the Artificial Intelligence National Excellence Program (grant no.: 2018-1.2.1-NKP-2018-00008).

References

[1] Aggarwal, C. C., and Zhai, C. A survey of text classification algorithms. In *Mining Text Data*, C. C. Aggarwal and C. Zhai, Eds. Springer US, Boston, MA, 2012, pp. 163–222.

[2] Farkas, R., Berend, G., Hegedűs, I., Kárpáti, A., and Krich, B. Automatic free-text-tagging of online news archives. In *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence* (Amsterdam, The Netherlands, The Netherlands, 2010), IOS Press, pp. 529–534.

[3] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. 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* (Valencia, Spain, 2017), ACL, pp. 427–431.

[4] Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., and Brown, D. E. Text classification algorithms: A survey. *ArXiv abs/1904.08067* (2019).

[5] Kudo, T. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Melbourne, Australia, 2018), ACL, pp. 66–75.

[6] Kudo, T., and Richardson, J. 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* (Brussels, Belgium, Nov. 2018), Association for Computational Linguistics, pp. 66–71.

[7] Sennrich, R., Haddow, B., and Birch, A. 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)* (Berlin, Germany, Aug. 2016), Association for Computational Linguistics, pp. 1715–1725.

[8] Szántó, Zs., Vincze, V., and Farkas, R. Magyar nyelvű szó- és karakterszintű szóbeágyazások [Word and character embeddings for Hungarian]. In *XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017) [13th Hungarian Conference on Computatinal Linguistics]* (Szeged, 2017), SZTE, pp. 323–328.

[9] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. XLNet: generalized autoregressive pretraining for language understanding. *CoRR abs/1906.08237* (2019).