Neural Machine Translation for Harmonized System Codes prediction

Xi Chen*
Computer Science Department, Open University of the Netherlands[1]

Stefano Bromuri†
Computer Science Department, Open University of the Netherlands[1]

Marko Van Eekelen‡
Computer Science Department, Open University of the Netherlands[1]

ABSTRACT

The harmonized system codes (HS codes) are used worldwide to categorize products in international shipments. In its basic form HS codes come in 6 digit format, subdivided hierarchically into groups of two digits (chapters, headings and subheadings). When shipping products, it is mandatory to specify a HS code for the purpose of producing a custom declaration. Currently the process is mostly carried out by human experts who take a decision on the HS code to be assigned to a shipment depending on the item description provided by the shipper. As such the process is time consuming and prone to errors due to generic, incomplete or non-interpretable descriptions. The objective of this research is to automate the classification of HS codes in order to increase productivity to cope with extra volume in the custom classification area. For the purpose of testing the developed models, we used an anonymized data set of shipments provided by DHL. The main contribution of this paper is we applied a deep learning model which have not been tried on tackling the HS code classification problem: an attention-based neural machine translation (NMT) model with integration of hierarchical loss. The model can classify around 29% percentage of the dataset where the model's accuracy can reach 85%.

CCS CONCEPTS

• **Computing methodologies**; • **Neural networks**; • **Machine translation**;

KEYWORDS

shipments, logistics, automatic classification, line items, hs codes

ACM Reference Format:
Xi Chen, Stefano Bromuri, and Marko Van Eekelen. 2021. Neural Machine Translation for Harmonized System Codes prediction. In *2021 6th International Conference on Machine Learning Technologies (ICMLT 2021), April 23–25, 2021, Jeju Island, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3468891.3468915

*Xi Chen : xi.chen3@dhl.com , xi.chen@ou.nl; Contact: +31657069134
†Stefano Bromuri: Stefano.Bromuri@ou.nl
‡Marko van Eekelen: marko.vaneekelen@ou.nl
[1]Vrije Universiteit Brussel Pleinlaan 2, gebouw B, lokaal 3B206, 1050 Bruxelles, Netherlands

ICMLT 2021, April 23–25, 2021, Jeju Island, Republic of Korea
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8940-2/21/04.
https://doi.org/10.1145/3468891.3468915

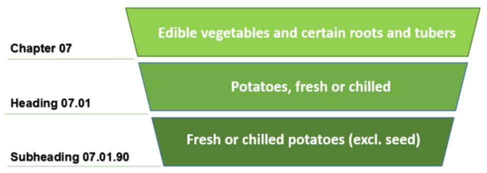
Chapter 07
Edible vegetables and certain roots and tubers
Heading 07.01
Potatoes, fresh or chilled
Subheading 07.01.90
Fresh or chilled potatoes (excl. seed)

Figure 1: HS Code example in six digits.

1 INTRODUCTION

The harmonized system codes (HS codes) are used worldwide to categorize products. In its basic form HS codes come in 6 digit format, subdivided hierarchically into groups of two digits (chapters, headings and subheadings). Figure 1 illustrates the structure of an HS code composed by six digits. When importing or exporting goods, it is mandatory to provide its associated HS code to the custom clearance. Currently, the majority of the work for assigning HS code was done by the domain expert. It is an intensive and error prone task. Thus, automatic classifying HS code is needed in order to provide aid to domain experts by proposing/predicting the HS Code. In this experiment, we are therefore analyzing the first six digits of the HS code which are standardized at a global level for all types of shipments. The goal is not looking for the model which has the best accuracy. Instead, we are looking for the model that has the best auto-classify rate/recall under certain accuracy requirements.

In the logistics sector, shipments have to be described by a number of attributes, such as origin, destination, shipper and item description of the items being shipped. As such, our hypothesis is that the problem of producing an HS code classification starting from item description and features of the shipment can be considered as a machine translation problem [1], where the HS code space represents the target language to be produced starting from the features and description of the line item being shipped. In addition, the codes take a hierarchical structure, with dependencies occurring between the chapters, headings and subheadings, therefore another possible modeling could imply a hierarchical classification model [2, 3].

Thanks to deep learning models such as the Neural Machine Translator (NMT) [1], machine translation has seen a quick advancement in recent years. NMT models rely on Recurrent Neural Networks (RNN) often in the form of long short term memories [4], and are usually modeled with an encoder-decoder architecture, where the encoder is fed with sentences of one origin language, plus additional features, and the decoder is fed with sentences of the target language.

Further advancements have been achieved thanks to the conception of attention models. Attention models allow to focus the

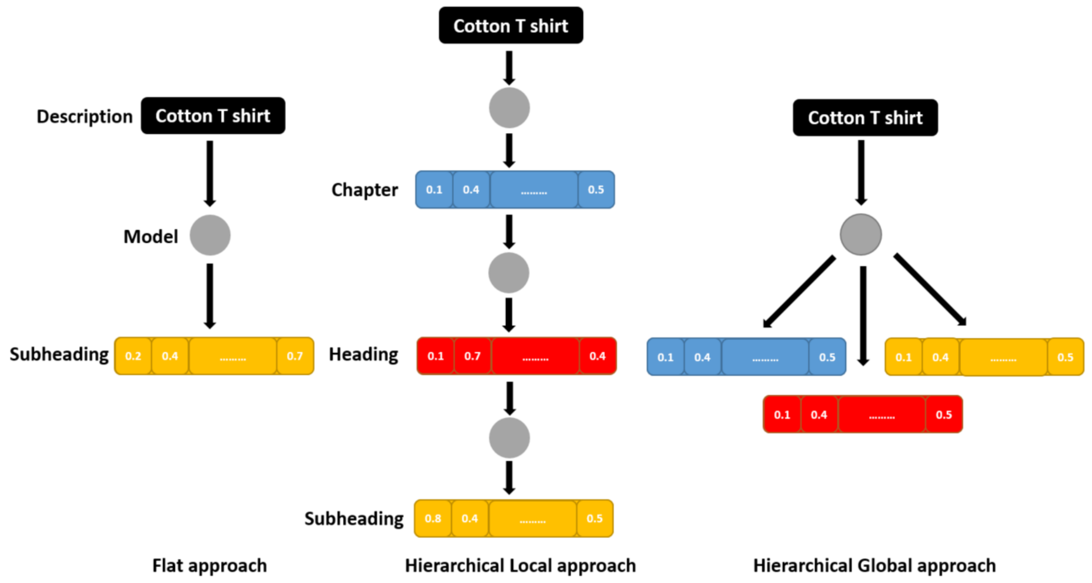

Figure 2: Hierarchical Model structure

network on parts of subparts of the sequence. They have shown to greatly improve tasks such as image captioning and language translation [5]. The problem of classifying HS codes has been previously recognized as a research problem [6], but it has been tackled using the exact word matching and ontologies, with limited ability to generalize to unseen descriptions. The main contribution of this paper with respect to the state of the art is to model the problem of classifying HS codes as a machine translation task, where the input language comprises the description of the shipment, its origin and destination and the output language is the HS code associated with the shipment. In addition to this, we also adapted hierarchical loss that allows us to improve over the basic NMT model. This is significant because the proposed model and hierarchical loss allow to classify automatically around 29% of the data used for the experimentation with an accuracy of 85% for codes comprising 6 digits which is *higher than the domain expert.*

The rest of the paper is structured as follows. Section 2 discusses relevant related work. Section 3 discusses the data sample used and Section 4 discusses the method applied in this contribution. Section 5 discusses the results concerning and the limitations of the study. Section 6 concludes this paper proposing potential future work directions.

2 RELATED WORK

Most of the work about HS codes concerns the definition of a knowledge base for a manual search of the code, given a shipment description, for example, Wei et al. [6] use an ontology based service to help a user generating the right code given a product. Singh [7] adopted the fuzzy logic to help identify wrongly classified HS code. Ding et al. follow a fuzzy logic approach by applying a background net to the automatic classification of HS codes [8], showing that a statistical approach can lead to better results than exact keywords matching.

Concerning classification, the HS code classification problem can be formulated as a hierarchical classification problem. There are three different approaches for hierarchical classification: flat approach, hierarchical local approach [2, 9] and hierarchical global [10, 11]. The flat approach addresses the hierarchical problem as a multi-class classification. The global local approach, a top-down structure like a tree, is specified in which each node requires a local classifier. Instead, the hierarchical global approach is utilizing one model and try to classify all at once. Figure 2 illustrates the difference between those three approaches.

The issue of the flat approach is that it ignores the hierarchical structure. The local approach [12, 13] only takes partial hierarchical information into account since all the local classifiers are isolated between each other. Recently, the global neural network based approach is prevailing when dealing with hierarchical classification tasks. This approach simulates the hierarchical structure by using neural network and modifying the loss function accordingly [14–18] in order to make sure the model will capture the information on global level.

Another field of research closely related to this problem is the one of multi-label learning [19]. Compared to standard classification approaches, in multi-label learning the items can have multiple labels at the same time. Several approaches exist to model the presence of multiple labels, such as for example binary relevance [20] chain classifiers [21], and also multi-label deep learning architectures [22, 23], Modeling the HS code automatic classification with a multi-label approach would imply defining an encoding for the chapter, heading, subheading sections of the HS code, that is not very practical due to the fact that each of these sections can in principle have many sub-labels (up to 100).

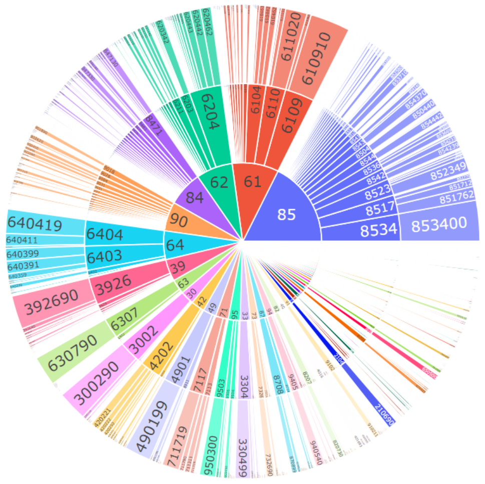

Figure 3: HS Code distribution

The NMT structure that being applied in this research itself has the advantage of carrying the hierarchical information while maintaining the label consistency.

3 DATA

The data used in this experiment develops in a period of eight months of shipments towards one country via the DHL network. The data set contains the following features: item description, origin, destination, origin airport, destination airport. It has 1.156 million records. Among those records, there are 476.128 (41.18%) unique descriptions, 705 origin airports, 40 destination airports, 183 export countries, 13.014 different combinations of origin airports and destination airports, 4.257 different HS code in six digits level and the distribution of it shows in Figure 3.

This work was prepared during the COVID-19 outbreak, so there is an unusual amount of masks and blood samples are shipped through the DHL network. Hence, the description that contained mask/kn95 and blood samples have been removed in order to make the conclusion more generalizable. The infrequent HS codes which appear less than 10 times also got removed in the cleaning step. Regarding the preprocessing of the text descriptions, we applied a standard NLP approach involving the conversion of every description and text field into lowercase, removed the punctuation and the digits. It is worth to mention that the data set is not clean. There are two main issues: first of all, often the description is not containing enough information to classify six digits HS code. Secondly, part of the HS codes are assigned wrongly due to human mistakes. In order to address those two issues a parallel work to this one is focusing on implementing a description quality measurement, but the details of this development are beyond the scope of this paper.

4 METHOD

Three different architectures are being tried out in this experiment: Hierarchical logistic regression, NMT and LSTM. The LSTM has an identical structure as NMT's encoder, the purpose is to identify the gain comes from the NMT decoder structure. We briefly describe these three architectures in this Section.

4.1 Logistic Regression

The first approach that is being applied is Hierarchical Multinomial logistic regression (HLR) [24]. It is a local hierarchical approach where we build a multinomial logistic regression model at each node as shown in Figure 2

For each multinomial logistic regression model, we expand on the node where it has the largest probability:

$$P(Y = K) = \frac{e^{\beta_i * X_i}}{\sum_{k=1}^{K} e^{\beta_k * X_i}}$$

K is the total possible output, β is the coefficient [24].

4.2 Neural Machine Translation

A Neural Machine Translator (NMT) is an encoder-decoder model for sequences as shown in Figure 4, and it is meant to translate one sequence (i.e. an English sentence) to another sequence (i.e. the respective French sentence). Formally, the NMT transforms sequences of vectors $x = (x_1, x_2, \ldots, x_n)$ into sequences of vectors $y = (y_1, y_2, \ldots, y_m)$ where the sequences may not necessarily have the same length (i. e. n different from m). The translation is often performed using an RNN network (for example an LSTM), due to their ability of representing sequences. Other architectures are possible, for example a combination of RNN and CNN can also be used. In the case of the RNN, in the encoder part of the NMT architecture, the hidden state of the RNN at time t, h_t is used to define a context vector $c = q(f(h_1, \ldots, h_m))$,where f and q are non-linear functions. The context vector effectively encodes all the information of the sequence. The training of the decoder part of the network, that is usually also represented by means of an RNN, happens by predicting the next word y_t given the context vector c and all the previously predicted words $y_1, y_2, \ldots, y_{t-1}$.

Formally this can be written as:

$$P(y) = \prod_{t=1}^{T} p(y_t | y_1, y_2, \ldots, y_{t'-1}, c)$$

In terms of an RNN, this probability is expressed as:

$$P(y_t | y_1, y_2, \ldots, y_{t'-1}, c) = g(y_t - 1, s_t, c)$$

Where g is a nonlinear, potentially multi-layered, function that outputs the probability of y_t, and s_t is the hidden state of the RNN.

4.2.1 *Attention models.*

The context vector c of the NMT has the disadvantage of behaving like a bottleneck concerning the information contained in the sequence. Effectively speaking the fixed length of the vector reduces the ability of the NMT to remember long sequences, often forgetting important parts of the sequence. Attention models have been introduced to reduce this effect. This paper makes use of additive attention [25] that makes use of an explicit layer of neurons to reweigh the importance of certain parts of a sequence. Formally, the function of such a layer can be expressed as:

$$f_{att}(h_i, s_j) = v_a^T tanh\left(W_a\left[h_i; s_j\right]\right)$$

Meaning that additive attention learns to align hidden states of the decoder (s_j) with hidden states of the encoder (h_i). In doing so,

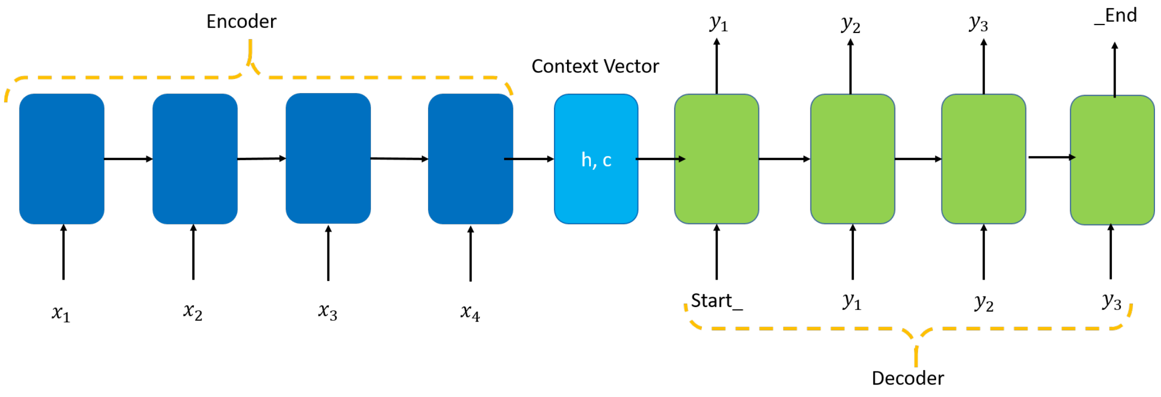

Figure 4: NMT Architecture

the attention layer also learns parameters v_a and W_a . The final scores of the alignment are calculated by means of a softmax layer that reweighs the importance of the hidden states in the prediction.

4.2.2 Embedding. The embedding used the in this experiment is randomly initialized and trained together with the model. The traditional pre-trained embedding [26–28] has really low coverage in words since the description contains a lot of domain-specific vocabulary. Also, the majority of the descriptions do not have any grammar structure in them, so the contextual embedding [29–31] is not the first choice here. The contextual embedding and re-train traditional embedding are worth trying out in future work.

4.2.3 Hierarchical loss. The NMT model itself trained in a teaching force [32] way. It could argue that it might not necessarily learn the global information considering it evaluates the model in the single output level instead of in the sequence level. So the idea of hierarchical loss [15] is also being introduced for addressing the hierarchical classification, which is an integration of local and global loss. The loss function is defined as follow:

$$L(y,\hat{y}) = \alpha * \sum y_k * log(\hat{y}_k) + \beta * H$$

The α and β are the hyper-parameters, y_k and $\hat{y}_k$ are true and predicted value at the k digits respectively. H is the binary value, it equals to 0 if the whole sequence is predicted correctly, 1 otherwise.

4.3 Long Short Term Memory

In order to identify the gain from the NMT decoder structure, we also tested an LSTM model that has the same structure as NMT encoder. The structure of the model is shown in Figure 5. The LSTM cell will take one input at a time, then it will predict the chapter, heading, subheading at one goal after it went through all the input.

The LSTM is one type of RNN, and it was introduced in order to solve the vanishing gradient issue that vanilla RNN suffered. In LSTM cell, it contains three different gates to control the information flow. They are: input gate i_t, forget gate f_t and output gate o_t respectively. Each gate state is determined by its internal states: weights W and bias b, output from the previous state h_{t-1}, and the current input x_t.

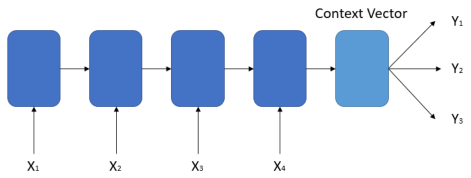

Figure 5: LSTM based model

The way it updates the state:

$$i_t = sigmoid(W_{ix} * x_t + W_{ih} * h_{t-1} + b_i)$$

$$f_t = sigmoid\left(W_{fx} * x_t + W_{fh} * h_{t-1} + b_f\right)$$

$$o_t = sigmoid(W_{ox} * x_t + W_{oh} * h_{t-1} + b_o)$$

The internal state will also get updated during each step:

$$C_t = f_t' * C_{t-1} + i_t * tanh(W_c * x_t + b_c)$$

The output at each step is:

$$h_t = o_t * tanh(c_t)$$

5 RESULTS

In this experiment we evaluated five different models: HLR, LSTM, LSTM with hierarchical loss (LSTM-HL), NMT and NMT with hierarchical loss (NMT-HL). The results are compared in two dimensions: the recall under certain requirements and the accuracy on that scope.

The data $X = \{x_1, x_2, \ldots, x_n\}$ is analyzed with respect to a confidence score $P = \{p_1, p_2, \ldots, p_n\}$ where p_n is the model's predicted probability for the data point x_n. Empirically that high confidence score produce high precision at identifying correct classified examples [33]. So a threshold T is applied on the confidence score p_n to obtain a certain desired accuracy. The subset of the X'_{val} is the recall, where $X'_{val} \subseteq X_{val}$ and its $P'_{val} \geq T$. The threshold is calculated on the validation dataset and the result is evaluated on the test dataset by applying the same threshold. In order to make the result closer to reality, we split the data temporally, using all

Table 1: Model performance comparison.

AccuracyThreshold	HLR		NMT		NMT-HL		LSTM		LSTM-HL	
	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall
N.A.	**45.95**	100	45.00	100	45.42	100	43.08	100	41.94	100
70	69.40	52.95	70.44	53.53	70.24	**55.50**	**70.72**	54.05	69.76	53.04
75	74.54	44.12	75.52	45.44	**75.65**	**47.26**	75.51	46.30	74.56	44.69
80	79.69	35.43	80.62	37.61	80.70	**39.15**	80.45	38.74	**80.76**	36.83
85	85.15	27.09	85.41	29.83	**86.09**	29.52	85.03	**30.38**	85.17	29.58
90	91.06	18.40	90.57	**21.20**	**91.35**	20.39	90.95	20.96	91.13	21.13

the data, except for the last two months, for training, and using the last two month of the data for validation and testing respectively.

5.1 Result analysis

The result is given in Table 1. If no limitation (N.A) is imposed, HLR has the best performance among all models. The NMT-HL performs the best recall when the wanted accuracy is at 70%, 75% and 80%. The deep learning model has more or less the same accuracy and recall at the threshold on 85% and 90%. In general, the deep learning model has better performance compared to HLR model at the accuracy threshold 70% or above.

The additional hierarchical loss is improving the NMT model's performance, but it deteriorates the result of the LSTM model. One explanation could be without hierarchical structure or direct connection in the predicted chapter, heading and subheading, the additional hierarchical penalty might just confuse the model. The overall difference on those five models are not big, this could come from below reasons: first, the data is noisy (i.e. wrong chapters, headings and subheadings), and this might have an impact on the models; secondly, the majority of the descriptions are short and do not have language structure, this could limiting the effectiveness of RNN networks. Also, the dataset is not that large considering it has 4257 different HS code combinations, the deep learning model might benefit more if more data are provided.

6 CONCLUSION

A deep learning model NMT-HL algorithm has being applied to tackle the HS classification issue. We can auto-classify around 29% of the descriptions in the current data set when 85% accuracy is desired. It can be used to improve the DHL agents productivity and accuracy. In terms of future work, on the business side, the architecture will be extended to predict more than six digits which might have different output length. Then other questions can be further investigated: from the standpoint of embeddings, we can fine-tune the pre-trained embedding on the current data instead of training it from scratch, so that we can utilize more advanced embedding[30, 34]. A transformer network [5] has been already attempted by the authors, but no **automation rate** improvement compared to **NMT** could be found, albeit the detailed analysis and fine-tuning still need to take place. A potential model to try is a Transformer-based pre-trained seq2seq model [35, 36]. Also, the positional encoding [5, 37] that is being applied on most of the latest seq2seq models [38, 39] might have some impact if added to the decoding part on NMT model. Finally, we could also investigate further the probabilistic output of the model and see the alternative in determining whether or not we should trust the model's prediction [33, 40].

REFERENCES

[1] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with 245 neural networks, in: Advances in neural information processing systems, 2014, pp. 3104–3112

[2] A. Sun, E.-P. Lim, Hierarchical text classification and evaluation, in: Proceedings 2001 IEEE International Conference on Data Mining, IEEE, 2001, pp. 521–528

[3] C. N. Silla, A. A. Freitas, A survey of hierarchical classification across different application domains, Data Mining and Knowledge Discovery 22 (1- 2) (2011) 31–72.

[4] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, 13 L. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.

[6] X. Wei, L. Yinsheng, X. Yingxiao, M. Zhanxin, Implementing knowledge base for hs matchmaking, in: 2006 IEEE International Conference on e260 Business Engineering (ICEBE'06), IEEE, 2006, pp. 579–584.

[7] A. K. Singh, R. Sahu, Decision support system for hs classification of commodities, in: Proceedings of the 2004 IFIP International Conference on Decision Support Systems (DSS 2004), 2004

[8] L. Ding, Z. Fan, D. Chen, Auto-categorization of hs code using background 265 net approach, Procedia Computer Science 60 (2015) 1462–1471.

[9] D. Koller, M. Sahami, Hierarchically classifying documents using very few words, Tech. rep., Stanford InfoLab (1997).

[10] C. N. Silla Jr, A. A. Freitas, A global-model naive bayes approach to the hierarchical prediction of protein functions, in: 2009 Ninth IEEE Interna270 tional Conference on Data Mining, IEEE, 2009, pp. 992–997.

[11] S. Kiritchenko, S. Matwin, A. F. Famili, *et al.*, Functional annotation of genes using hierarchical text categorization, in: Proc. of the ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics, 2005.

[12] L. Cai, T. Hofmann, Hierarchical document categorization with support vector machines, in: Proceedings of the thirteenth ACM international conference on Information and knowledge management, 2004, pp. 78–87.

[13] N. Cesa-Bianchi, C. Gentile, L. Zaniboni, Hierarchical classification: combining bayes with svm, in: Proceedings of the 23rd international conference 280 on Machine learning, 2006, pp. 177–184.

[14] Z. Wu, S. Saito, Hinet: Hierarchical classification with neural network, arXiv preprint arXiv:1705.11105 (2017).

[15] D. Gao, W. Yang, H. Zhou, Y. Wei, Y. Hu, H. Wang, Deep hierarchical classification for category prediction in e-commerce system, arXiv (2020) 285 arXiv–2005.

[16] J. Wehrmann, R. Cerri, R. Barros, Hierarchical multi-label classification networks, in: International Conference on Machine Learning, 2018, pp. 5075–5084.

[17] K. Kowsari, D. E. Brown, M. Heidarysafa, K. J. Meimandi, M. S. Gerber, 290 L. E. Barnes, Hdltex: Hierarchical deep learning for text classification, in: 2017 16th IEEE international conference on machine learning and applications (ICMLA), IEEE, 2017, pp. 364–371.

[18] Y. Mao, J. Tian, J. Han, X. Ren, Hierarchical text classification with reinforced label assignment, arXiv preprint arXiv:1908.10419 (2019).

[19] F. Charte, A comprehensive and didactic review on multilabel learning software tools, IEEE Access 8 (2020) 50330–50354.

[20]] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, X. Geng, Binary relevance for multilabel learning: an overview, Frontiers of Computer Science 12 (2) (2018) 191–202

[21] X. Jun, Y. Lu, Z. Lei, D. Guolun, Conditional entropy based classifier chains for multi-label classification, Neurocomputing 335 (2019) 185–194.

[22] J. Nam, E. L. Menc´ıa, H. J. Kim, J. F¨urnkranz, Maximizing subset accuracy with recurrent neural networks in multi-label classification, in: Advances in neural information processing systems, 2017, pp. 5413–5423.
[23] V. Umaashankar, G. S. S, Multi-label multi-class hierarchical classification using convolutional seq2seq., in: KONVENS, 2019.
[24] D. B¨ohning, Multinomial logistic regression algorithm, Annals of the institute of Statistical Mathematics 44 (1) (1992) 197–200.
[25] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly 310 learning to align and translate, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL http://arxiv.org/abs/1409.0473
[26] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word 315 representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543
[27] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in neural information processing systems, 2013, pp. 3111–3119.
[28] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, Enriching word vectors with subword information, Transactions of the Association for Computational Linguistics 5 (2017) 135–146.
[29] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, arXiv preprint 325 arXiv:1802.05365 (2018).
[30] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).
[31] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language 330 models are unsupervised multitask learners, OpenAI blog 1 (8) (2019) 9
[32] R. J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural computation 1 (2) (1989) 270–280.
[33] H. Jiang, B. Kim, M. Guan, M. Gupta, To trust or not to trust a classifier, Advances in neural information processing systems 31 (2018) 5541–5552.
[34] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, 335 in: Advances in neural information processing systems, 2019, pp. 5753– 5763.
[35] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, H.-W. Hon, Unified language model pre-training for natural language understanding and generation, in: Advances in Neural Information Processing 340 Systems, 2019, pp. 13063–13075.
[36] K. Song, X. Tan, T. Qin, J. Lu, T.-Y. Liu, Mass: Masked sequence to sequence pre-training for language generation, arXiv preprint arXiv:1905.02450 (2019).
[37] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. N. Dauphin, Convolutional 345 sequence to sequence learning, arXiv preprint arXiv:1705.03122 (2017).
[38] P. Shaw, J. Uszkoreit, A. Vaswani, Self-attention with relative position representations, arXiv preprint arXiv:1803.02155 (2018).
[39] S. Takase, N. Okazaki, Positional encoding to control output sequence length, arXiv preprint arXiv:1904.07418 (2019).
[40] A. Amini, W. Schwarting, A. Soleimany, D. Rus, Deep evidential regression, Advances in Neural Information Processing Systems 33 (2020).