TeamX at CLEF eHealth 2020: ICD Coding with N-gram Encoder and Code-filtering Strategy

Yuki Tagawa, Norihisa Nakano, Ryota Ozaki,
Tomoki Taniguchi and Tomoko Ohkuma

Fuji Xerox Co., Ltd, Japan
{tagawa.yuki,nakano.norihisa,ryota.ozaki,Taniguchi.Tomoki,Ohkuma.Tomoko}
@fujixerox.co.jp

Abstract. The International Classification of Diseases (ICD) is a medical classification that provides a systematized code of diseases. ICD is widely used for statistical comparisons and patient billing; however, manual ICD coding is time-consuming and prone to errors. In this study, we work on an automatic ICD10-CM and ICD10-PCS coding to Spanish clinical cases at CLEF eHealth 2020 Task 1.
We tackle the ICD10-CM and ICD10-PCS coding as a multi-label classification problem and our method has three main aspects: (i) *N-gram encoder*: learning N-gram embeddings by encoding an input document; (ii) *Code-filtering strategy*: reducing the label space by limiting the number of target code; (iii) *Weighted binary cross-entropy (BCE)*: extending the BCE to alleviate the data imbalance problem.
We evaluated our method based on the mean average precision, achieving final scores of 0.299 for ICD10-CM and 0.199 for ICD10-PCS.

1 Introduction

In clinical practice, considerable amounts of text data (e.g., discharge summaries, radiology reports, and other narrative components of electronic health records) are created every day. Such data are managed using the International Classification of Diseases (ICD) codes for reporting diagnosis and statistical comparisons of morbidity and mortality. ICD is a medical classification provided by the World Health Organization, and it assigns a unique alphanumeric code to diseases, injuries, signs, procedures, and symptoms.

Although ICD codes are widely used for statistical analysis, decision-making, and even for reimbursement, manual ICD coding is time-consuming and prone to errors. Hence, automatic ICD coding is in high demand.

Automatic ICD coding [12, 16, 19] is the prediction of suitable ICD codes on the basis of an input document. As a type of multilingual information extraction, the CLEF eHealth community has been organizing shared tasks on ICD coding since 2016. Furthermore, several methods have been proposed using topic modeling [4], pattern matching [13, 11], information retrieval-style ranking [20], sequence-to-sequence (seq2seq) [9, 3, 15], and bidirectional encoder representations from transformer (BERT)-based models [2, 18].

In this paper, we describe the approach of TeamX for the ICD10-CM [1] and ICD10-PCS [2] coding to Spanish clinical cases at CLEF eHealth 2020 Task 1 [6, 10]. The organizers prepared a CodiEsp corpus of 1,000 clinical cases in Spanish. This corpus was manually assigned ICD10-CM and ICD10-PCS codes by clinical coding professionals meeting strict quality criteria.

We found the following difficulties in the CLEF eHealth 2020 ICD coding task.

1. *The CodiEsp corpus has a large number of words per document*: In the CodiEsp corpus, the average number of words per document is approximately 396.2. In contrast, those in the CépiDc, KSH-HU, and ISTST-IT datasets [14] are 10.0, 7.9, and 46.0, respectively[3]. Compared with the other corpora in CLEF eHealth, the CodiEsp corpus has the largest number of words per document. In 2018, the seq2seq model [3] achieved the best performance. This model learns the document embedding by encoding sequences with a recurrent neural network (RNN) [17] and predicts the code sequences from this embedding. However, when processing long documents such as the CodiEsp Corpus, it is difficult to encode the documents into a single, fixed-size representation using an RNN or BERT [5].
2. *There is a large number of codes*: In general ICD coding, suitable codes must be predicted from a large number of codes (ICD10-CM and ICD10-PCS have approximately 87,000 and 98,000 types of codes for this task, respectively). In previous methods [2, 18], ICD coding was considered as a multi-label classification (MLC); however, it is usually difficult to learn a classification model with a large label space because the labels are highly imbalanced.

Considering the features mentioned above, we propose a model based on previous studies [16, 12]. Our method has three main aspects:

1. *N-gram encoder*: In the CodiEsp corpus, the number of words per document is large, and the ICD code is annotated into token N-grams. Therefore, we introduce an N-gram encoder to learn an N-gram representation rather than encoding the entire document into a single, fixed-size representation.
2. *Code-filtering strategy*: It is difficult to learn an ICD coding model as an MLC with a large label space. Therefore, we introduce a strategy to reduce the label space by limiting the number of target codes.

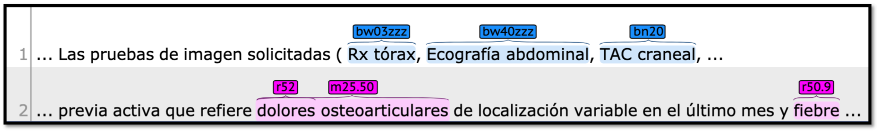
[1] `https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/ICD10CM/index.html`

[2] `https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/ICD10PCS/index.html`

[3] These datasets include death certificates consisting of term sequences in which an ICD10 code can be directly assigned to each term.

Table 1. Statistics of the dataset. Ground truth for the test set has not yet been published.

	Train	Development	Test
# of documents	500	250	250
# of tokens	174,509	88,074	88,178
# of total ICD10-CM codes	5,639	2,677	-
# of total ICD10-PCS codes	1,550	817	-
# of unique ICD10-CM codes	1,767	1,158	-
# of unique ICD10-PCS codes	563	375	-



... Las pruebas de imagen solicitadas (Rx tórax, Ecografía abdominal, TAC craneal, ...

... previa activa que refiere dolores osteoarticulares de localización variable en el último mes y fiebre ...

Fig. 1. Example of annotated data. The first sentence is assigned ICD10-PCS. The second sentence is assigned ICD10-CM. In the second sentence, r52 corresponds to "dolores" and m25.50 corresponds to "dolores osteoarticulares."

3. *Weighted binary cross-entropy (BCE)*: In previous studies on MLC, BCE was used as a loss function; however, for MLC with a large label space such as ICD coding, data imbalance must be avoided. To alleviate this problem, we extend the BCE by introducing a weight variable.

In the experiments, our method achieved mean average precision (MAP) scores of 0.299 and 0.199 for ICD10-CM and ICD10-PCS, respectively.

2 CodiEsp corpus

The CodiEsp corpus [10] comprises 1,000 clinical cases in Spanish and was interpreted by clinical coding professionals satisfying strict quality criteria. Table 1 lists the corpus statistics. Compared with previous ICD coding datasets in the CLEF eHealth, this corpus has a larger number of words per document and is annotated into a span of characters corresponding to the ICD code. Figure 1 illustrates an example of annotated data, where ICD codes are assigned to words or phrases that correspond to diseases and symptoms, etc.

3 Method

Figure 2 shows an overview of our model. We approach this task as an MLC. Our model mainly comprises an N-gram encoder, a code encoder, code-wise attention module, weighted BCE, and code-filtering strategies. In the following subsections, we describe each component of the model.

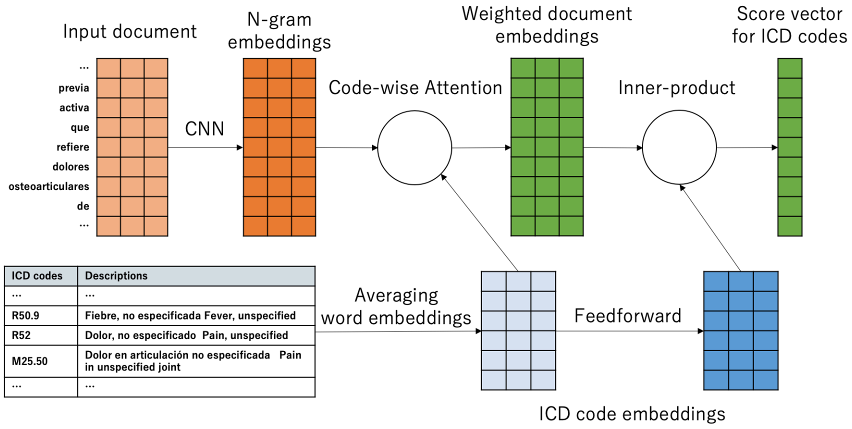

Input document
N-gram embeddings
Weighted document embeddings
Score vector for ICD codes
...
previa
activa
que
refiere
dolores
osteoarticulares
de
...
CNN
Code-wise Attention
Inner-product
ICD codes
Descriptions
...
...
R50.9
Fiebre, no especificada Fever, unspecified
R52
Dolor, no especificado Pain, unspecified
M25.50
Dolor en articulación no especificada Pain in unspecified joint
...
...
Averaging word embeddings
Feedforward
ICD code embeddings

Fig. 2. Overview of our model.

3.1 N-gram encoder

In the annotated texts, the ICD codes correspond to words or phrases that denote diseases, injuries, signs, symptoms, or procedures. We assume that N-gram representations in an input document are effective features for code prediction. An N-gram encoder learns N-gram embeddings from an input document using a convolutional neural network [7] (CNN):

$$\boldsymbol{D_1} = \mathrm{CNN}(\boldsymbol{X}), \tag{1}$$

where $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ is a trainable document feature matrix initialized with pre-trained word embeddings, n is the number of words in the input document, and d is the embedding size of the words. Moreover, CNN returns an N-gram feature matrix $\boldsymbol{D_1} \in \mathbb{R}^{(n-s+1) \times u}$, where s is the window size of the convolution filters (the size of the word-level N-gram), and u is the number of convolution filters. Each column of $\boldsymbol{D_1}$ represents an N-gram features.

3.2 Code encoder

We utilize the code descriptions to learn the code embeddings. The code embedding matrix, $\boldsymbol{L_2} \in \mathbb{R}^{C \times d}$, is computed as follows:

$$\boldsymbol{L_2} = \tanh(\mathrm{dropout}(\boldsymbol{L_1})\boldsymbol{W_1} + \boldsymbol{b_1}). \tag{2}$$

The initial code embedding, $\boldsymbol{L_1} \in \mathbb{R}^{C \times d}$, is computed by averaging the pre-trained embeddings of the words in the descriptions, where C is the total number of ICD codes. Here, $W_1 \in \mathbb{R}^{d \times d}$ and $b_1 \in \mathbb{R}^{d}$ are trainable parameters.

3.3 Code-wise attention

We introduce a code-wise attention mechanism that learns the relevance between N-grams and ICD codes. First, we pass the N-gram features, $\boldsymbol{D_1}$, to a feed-forward network as follows:

$$\boldsymbol{D_2} = \tanh(\boldsymbol{D_1}\boldsymbol{W_2} + \boldsymbol{b_2}), \tag{3}$$

where $W_2 \in \mathbb{R}^{u \times d}$ and $b_2 \in \mathbb{R}^d$ are trainable parameters. Next, we calculate the code-wise attention matrix:

$$\boldsymbol{A} = \mathrm{softmax}(\boldsymbol{D_2}\boldsymbol{L_1}^{\mathrm{T}}), \tag{4}$$

where $\boldsymbol{A} \in \mathbb{R}^{(n-s+1) \times C}$ represents the relevance between each N-gram and each ICD code. Finally, we calculate a weighted document feature matrix, $\boldsymbol{D_3} \in \mathbb{R}^{C \times d}$, and a score vector, $\boldsymbol{Y} \in \mathbb{R}^C$, for each ICD code:

$$\boldsymbol{D_3} = \mathrm{relu}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{D_2}), \tag{5}$$

$$\boldsymbol{Y} = \mathrm{sigmoid}(\boldsymbol{D_3} \cdot \boldsymbol{L_2}^{\mathrm{T}}), \tag{6}$$

where $\cdot$ denotes the inner product.

In the testing phase, our model ranks the ICD codes using the predicted scores and removes the codes with a score below pre-defined threshold t from the ranking.

3.4 Weighted BCE

We consider the ICD coding as an MLC and therefore naturally use the BCE as a loss function. However, we should be careful when the numbers of positive and negative samples in the ground truth $\hat{\boldsymbol{Y}}$ are significantly imbalanced because there are many types of ICD codes. If the MLC model is trained using the standard BCE, the trained model predicts a low score for all the codes because the elements of $\hat{\boldsymbol{Y}}$ are almost zero (negative). To alleviate this problem, we introduce a weight variable, w_p, for a positive sample into the BCE. We train our model using weighted BCE as a loss function:

$$\mathrm{Loss} = -\frac{1}{C}\sum_{i=1}^{C} w_p y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i}), \tag{7}$$

$$w_p = \frac{\text{NEGATIVE COUNTS}}{\text{POSITIVE COUNTS}}, \tag{8}$$

where $\hat{y_i} \in \{0, 1\}$ and $y_i \in [0, 1]$ are the ground truth and predicted score for the i-th code, respectively. POSITIVE COUNTS represents the number of elements with a value of 1 in $\hat{\boldsymbol{Y}}$, and NEGATIVE COUNTS is the number of elements with a value of 0 in $\hat{\boldsymbol{Y}}$. The weighted BCE returns a higher loss value than the standard BCE if the model predicts a low score for the appropriate ICD code during training.

Table 2. Number of ICD codes in each dataset applying code-filtering strategies. *ORIGINAL* is a strategy using which nothing is removed.

	ORIGINAL	*AND*	*OR*
ICD10-PCS	87,170	211	727
ICD10-CM	98,288	731	2,194

Table 3. List of hyperparameters of our model.

Batch size	4
Optimizer	Adam (beta1 = 0.9, beta2 = 0.999)
Learning rate	0.0001
Pre-trained word embeddings size d	300
The number of convolution filters u	300
Dropout rate	0.2

3.5 Code-filtering strategy

In this task, the model must predict suitable codes from the input document. It is usually difficult to learn a classification task with a large label space. Therefore, we apply two code-filtering strategies, *AND* and *OR*, to reduce this space. Here, *AND* is a strategy using which our model predicts only the ICD codes included in both the training and development sets, and *OR* is a strategy with which our model predicts only the ICD codes included in either the training or development set. Table 2 shows the number of ICD codes applied to each code-filtering strategy. The code size C depends on the strategy applied.

4 Experiments

4.1 Experimental settings

We implemented our model using PyTorch[4] and trained the model using a training set from the CodiEsp corpus[5], code descriptions[6], and pre-trained Spanish medical embeddings[7]. Table 3 lists the hyperparameters of our model.

As a baseline, we built a term frequency–inverse document frequency (TFIDF)-based method. First, as the baseline, the word-level TFIDF scores from the CodiEsp corpus and the code descriptions are calculated, and L2 normalization is then applied to each TFIDF vector. Second, the cosine similarity between the TFIDF vector of the input document and that of each code description is calculated. Finally, the codes with a similarity are ranked, and the codes with

[4] `https://pytorch.org/`

[5] `https://zenodo.org/record/3606662#.XwVLmZP7TOR`

[6] `https://zenodo.org/record/3706838#.XwVLTZP7TOQ`

[7] `https://zenodo.org/record/3626806#.XwKxx5P7TOR`

Table 4. Experimental results of ICD10-PCS coding. The best scores are highlighted in bold. We submitted the outputs of the marked (✓) models for the final evaluation.

Model configuration					MAP	
Model	N-gram size s	Filtering strategy	Threshold t	Loss function	Dev	Test
Ours (✓)	2-gram	*AND*	0.0	Weighted	**0.235**	**0.190**
Ours (✓)	2-gram	*AND*	0.5	Weighted	0.223	0.182
Ours (✓)	2-gram	*OR*	0.0	Weighted	0.185	0.166
Ours (✓)	2-gram	*OR*	0.5	Weighted	0.177	0.160
Ours (✓)	3-gram	*AND*	0.0	Weighted	0.216	0.186
Ours	2-gram	*AND*	0.0	Standard	0.130	-
Ours	3-gram	*AND*	0.5	Weighted	0.202	-
Ours	3-gram	*OR*	0.0	Weighted	0.196	-
Ours	3-gram	*OR*	0.5	Weighted	0.188	-

Table 5. Experimental results of ICD10-CM coding. The best scores are highlighted in bold. We submitted the outputs of the marked (✓) models for the final evaluation.

Model configuration					MAP	
Model	N-gram size s	Filtering strategy	Threshold t	Loss function	Dev	Test
Ours (✓)	2-gram	*AND*	0.0	Weighted	**0.290**	**0.299**
Ours (✓)	2-gram	*AND*	0.5	Weighted	0.272	0.284
Ours (✓)	2-gram	*OR*	0.0	Weighted	0.240	0.265
Ours (✓)	2-gram	*OR*	0.5	Weighted	0.232	0.259
Ours	2-gram	*AND*	0.0	Standard	0.094	-
Ours	3-gram	*AND*	0.0	Weighted	0.280	-
Ours	3-gram	*AND*	0.5	Weighted	0.261	-
Ours	3-gram	*OR*	0.0	Weighted	0.230	-
Ours	3-gram	*OR*	0.5	Weighted	0.221	-
Baseline(✓)	1-gram	*AND*	0.0	-	0.068	0.065
Baseline	1-gram	*OR*	0.0	-	0.023	-

a similarity below pre-defined threshold t are removed. We used StanfordNLP[8] and scikit-learn[9] to calculate the TFIDF score.

4.2 Results and discussion

We trained the model for 100 epochs and selected the best model of the development data for the testing. We used MAP to evaluate the model.

Tables 4 and 5 show the experimental results of the ICD10-PCS and ICD10-CM codes, respectively. Our method outperforms the TFIDF-based method as a baseline. The MAP of the *AND* strategy is higher than that of the *OR* strategy

[8] `https://stanfordnlp.github.io/stanfordnlp/`

[9] `https://scikit-learn.org/stable/`

in both the ICD10-PCS and ICD10-CM codes. It can be seen that the strategy of limiting the target code is effective for this task. As a future study, we are also interested in a frequency-based or MAP-maximizing strategy.

Comparing the weighted BCE and the standard BCE setting, the weighted BCE is more effective. In particular, we observed a large elongation in the ICD10-CM dataset (Table 5). Because the ICD10-CM dataset has a larger number of codes even with the *AND* strategy (Table 2) and exhibits a higher data imbalance, as described in Section 3.4, the weighted BCE proved to be effective.

5 Conclusion

We addressed the automatic coding of the ICD10-CM and ICD10-PCM for Spanish clinical cases at CLEF eHealth 2020 Task 1. We considered the ICD coding as an MLC, and our method had three main aspects: (i) *N-gram encoder*: learning N-gram embeddings by encoding an input document; (ii) *Code-filtering strategy*: reducing the label space by limiting the number of target codes; (iii) *Weighted BCE*: extending the BCE to alleviate the data imbalance problem.

Our method achieved MAP scores of 0.299 and 0.199 for the ICD10-CM and ICD10-PCS datasets, respectively. In particular, we confirmed the effectiveness of both the code-filtering strategies, *AND* and *OR*, and the weighted BCE as a loss function.

In future studies, to improve the performance, we plan to apply data argumentation using back-translation [2] and integrate the BERT in the clinical domain [1, 8] into a CNN encoder.

References

1. Alsentzer, E., Murphy, J., Boag, W., Weng, W.H., Jin, D., Naumann, T., McDermott, M.: Publicly available clinical BERT embeddings. In: Proceedings of Clinical NLP Workshop (2019)
2. Amin, S., Neumann, G., Dunfield, K., Vechkaeva, A., Chapman, K.A., Wixted, M.K.: Mlt-dfki at clef ehealth 2019: Multi-label classification of ICD-10 codes with bert. In: CLEF 2019 Online Working Notes (2019)
3. Atutxa, A., Casillas, A., Ezeiza, N., Fresno, V., Goenaga, I., Gojenola, K., Martínez, R., Anchordoqui, M.O., Perez-De-Viñaspre, O.: Ixamed at clef ehealth 2018 task 1: ICD10 coding with a sequence-to-sequence approach. In: CLEF 2018 Online Working Notes (2018)
4. Dermouche, M., Looten, V., Flicoteaux, R., Chevret, S., Velcin, J., Taright, N.: Ecstra-inserm @ clef ehealth2016-task 2: ICD10 code extraction from death certificates. In: CLEF 2016 Online Working Notes (2016)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL (2019)
6. Goeuriot, L., Suominen, H., Kelly, L., Miranda-Escalada, A., Krallinger, M., Liu, Z., Pasi, G., Saez Gonzales, G., Viviani, M., Xu, C.: Overview of the CLEF eHealth evaluation lab 2020. In: Arampatzis, A., Kanoulas, E., Tsikrika, T., Vrochidis, S.,

Joho, H., Lioma, C., Eickhoff, C., Névéol, A., Cappellato, L., Ferro, N. (eds.) Experimental IR Meets Multilinguality, Multimodality, and Interaction: Proceedings of the Eleventh International Conference of the CLEF Association (CLEF 2020). LNCS Volume number: 12260 (2020)

7. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
8. Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C.H., Kang, J.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics (2019)
9. Miftahutdinov, Z., Tutubalina, E.: Kfu at clef ehealth 2017 task1: ICD-10 coding of english death certificates with recurrent neural networks. In: CLEF 2017 Online Working Notes (2017)
10. Miranda-Escalada, A., Gonzalez-Agirre, A., Armengol-Estapé, J., Krallinger, M.: Overview of automatic clinical coding: annotations, guidelines, and solutions for non-english clinical cases at codiesp track of CLEF eHealth 2020. In: Working Notes of Conference and Labs of the Evaluation (CLEF) Forum. CEUR Workshop Proceedings (2020)
11. Mottin, L., Gobeill, J., Mottaz, A., Pasche, E., Gaudinat, A., Ruch, P.: Bitem at clef ehealth evaluation lab 2016 task 2: Multilingual information extraction. In: CLEF 2016 Online Working Notes (2016)
12. Mullenbach, J., Wiegreffe, S., Duke, J., Sun, J., Eisenstein, J.: Explainable prediction of medical codes from clinical text. In: Proceedings of NAACL (2018)
13. van Mulligen, E.M., Afzal, Z., Akhondi, S.A., Vo, D., Kors, J.A.: Erasmus mc at clef ehealth 2016: Concept recognition and coding in french texts. In: CLEF 2016 Online Working Notes (2016)
14. Névéol, A., Robert, A., Grippo, F., Morgand, C., Orsi, C., Pelikan, L., Ramadier, L., Grégoire Rey, P.Z.: Clef ehealth 2018 multilingual information extraction task overview: ICD10 coding of death certificates in french, hungarian and italian. In: CLEF 2018 Online Working Notes (2018)
15. Réby, K., Cossin, S., Bordea, G., Diallo, G.: Sitis-isped in clef ehealth 2018 task 1: ICD10 coding using deep learning. In: CLEF 2018 Online Working Notes (2018)
16. Rios, A., Kavuluru, R.: Few-shot and zero-shot multi-label learning for structured label spaces. In: Proceedings of EMNLP (2018)
17. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. nature **323**(6088), 533–536 (1986)
18. Sänger, M., Weber, L., Kittner, M., Leser, U.: Classifying german animal experiment summaries with multi-lingual bert at clef ehealth 2019 task 1. In: CLEF 2019 Online Working Notes (2019)
19. Song, C., Zhang, S., Sadoughi, N., Xie, P., Xing, E.: Generalized zero-shot icd coding. CoRR (2019)
20. Zweigenbaum, P., Lavergne, T.: Limsi ICD10 coding experiments on cépidc death certificate statements. In: CLEF 2016 Online Working Notes (2016)