Divide and Conquer: An Extreme Multi-Label Classification Approach for Coding Diseases and Procedures in Spanish

Jose Barros
DCC, University of Chile
jose.barros.s@ug.uchile.cl

Matías Rojas
CMM, University of Chile
matias.rojas.g@ug.uchile.cl

Andrés Abeliuk
DCC, University of Chile
aabeliuk@dcc.uchile.cl

Jocelyn Dunstan
CMM, University of Chile
jdunstan@uchile.cl

Abstract

Clinical coding is the task of transforming medical documents into structured codes following a standard ontology. Since these terminologies are composed of hundreds of codes, this problem can be considered an Extreme Multi-label Classification task. This paper proposes a novel neural network-based architecture for clinical coding. First, we take full advantage of the hierarchical nature of ontologies to create clusters based on semantic relations. Then, we use a *Matcher* module to assign the probability of documents belonging to each cluster. Finally, the *Ranker* calculates the probability of each code considering only the documents in the cluster. This division allows a fine-grained differentiation within the cluster, which cannot be addressed using a single classifier. In addition, since most of the previous work has focused on solving this task in English, we conducted our experiments on three clinical coding corpora in Spanish. The experimental results demonstrate the effectiveness of our model, achieving state-of-the-art results on two of the three datasets. Specifically, we outperformed previous models on two subtasks of the CodiEsp shared task: CodiEsp-D (diseases) and CodiEsp-P (procedures). Automatic coding can profoundly impact healthcare by structuring critical information written in free text in electronic health records.

1 Introduction

The International Classification of Diseases (ICD) is a medical glossary published by the World Health Organization, which establishes specific coding rules for healthcare procedures and diseases. Mapping electronic health records into alphanumeric codes allows rapid summarization of information, which is necessary to calculate costs, support clinical decisions, and conduct detailed epidemiological studies. However, manual coding is time-consuming, resource-intensive, and error-prone, even for specialists. For this reason, developing tools to support this task is precious.

Extreme multi-label classification is a subset of the multi-label classification task in which the objective is to learn feature architectures and classifiers that can automatically tag a data point with the most relevant labels from a huge label set (Bhatia et al., 2016). The term extreme is justified in this case since the space of possible labels is generally very large and can exceed the number of documents in a given corpus.

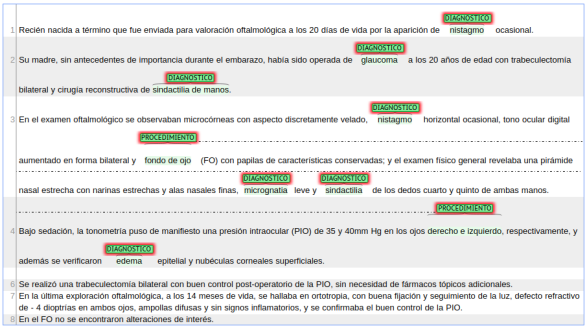

Figure 1: Example of a CodiEsp Electronic Health Record annotated, every diagnostic and procedure mention has a unique code. Every code from this Electronic Health Record is aggregated, and the document is labeled with all the codes present in the document. Each entity mention and its span is later used in the different data augmentation techniques explained in 3.6. Figure extracted from (Miranda-Escalada et al., 2020b).

Clinical coding is an important Natural Language Processing (NLP) task that seeks to automatically assign codes to medical documents following a standard terminology, such as the ICD glossary. Since each document can be labeled with more than one code from an extensive list, this task can be considered an Extreme Multi-label Classification task (Liu et al., 2017). An example of a Codiesp-D medical document is shown in Figure 1. Despite the availability of clinical coding datasets in Spanish, such as CANTEMIST (Miranda-Escalada

Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI), pages 138 - 147

et al., 2020a) or CodiEsp (Miranda-Escalada et al., 2020b), the size of these resources is not yet comparable to that for the English language. For example, Codiesp has 1,000 clinical case reports, while the MIMIC-III dataset (Johnson et al., 2016) has 52.726 discharge summaries. This scarcity of data forces models in other languages to have different architectures than those trained on English datasets.

To fill this gap, we introduce a novel architecture for solving clinical coding on three Spanish clinical datasets. This architecture is composed of two modules: the Ranker and the Matcher. The first module calculates the probability of a document belonging to a cluster, while the second performs the classification of documents into codes. Each cluster is created previously by analyzing the ontology of each dataset. Our experimental results show the effectiveness of our model, achieving state-of-the-art performance on CodiEsp-D (diseases) and CodiEsp-P (procedures) according to the Mean Average Precision (MAP) and F_1 score.

2 Related Work

In recent years, there has been a growing interest from the NLP research community in studying the clinical coding task. Early work focuses on creating machine learning-based classifiers with heavy feature engineering (Larkey and Croft, 1995; Goldstein et al., 2007). However, as mentioned in Kaur et al. (2021) and Teng et al. (2022), recent deep learning advancements have greatly improved clinical coding models' performance for all languages.

Regarding extreme multi-label architectures, there is one work that heavily inspired this work, which is X-Transformers (Chang et al., 2020). It proposes creating a clusterization of labels using the distance between the label descriptions encoded using contextualized embeddings retrieved from transformers' language models. Then they predict the clusters using a transformers classifier, and finally, they predict the labels over the subset of predicted clusters using one-vs-rest linear classifiers. The design of this architecture was thought to handle corpora much larger than the ones we have studied in this work, thus prioritizing time efficiency much more.

One of the most popular datasets used for clinical coding for the Spanish language is CodiEsp (Miranda-Escalada et al., 2020b). Most of the work proposed formulated the problem as text classification. In López-Garcıa et al. (2020), they used a transformer-based model to classify the sentences of the documents. Then, the whole document set of codes is the union of the sentence-level codes. Other approaches focused on solving the problem as a Named Entity Recognition (NER) task. In Cossin and Jouhet (2020), they created a dictionary based on entity mentions and code definitions. Then, they matched spans of documents with the code definitions in the dictionary using a tree-based algorithm. Finally, other ensemble-based models combined text classification and NER tasks to solve the clinical coding problem. For example, Blanco et al. (2020) implemented a model that used string-matching encoders and one-vs-rest document classification. This model obtained the best results in the competition.

Another important task of clinical coding in Spanish is Cantemist (Miranda-Escalada et al., 2020a), which aims to identify codes present in cancer diagnoses. This task had two winner systems obtaining the same MAP score. The first model proposed by García-Pablos et al. (2020) used different transformer-based models to predict specific parts of a code. These models were ensembled using a novel voting system. The second winner was López-García et al. (2020), who reused their approach proposed in CodiEsp but further pre-trained a language model with a private oncology corpus.

Recent work by López-García et al. (2021) outperformed previous models in CodiEsp and CANTEMIST by a wide margin. First, they trained three multilingual language models using private oncology datasets and then fine-tuned these models for classifying documents into codes. To improve the performance of their models, they ensembled the results from five different instances of each trained transformer.

3 Model

Our proposed architecture comprises two main modules: the Matcher and the Ranker. The first module calculates the probability that a document belongs to some cluster, while the second one calculates the probability of codes in the document. The results of both modules are used to perform the final prediction of codes. This process is carried out by multiplying the probability of codes obtained from the Ranker for each document with the code cluster probability obtained from the Matcher module. We refer to this approach as the Divide and

emc

Code partitioning	Assigned Cluster
From a00 to b99	Some diseases caused by infections and parasites
From c00 to d49	Tumors and neoplasia
From d50 to d89	Diseases of the blood and hematopoietic organs
From h00 to h59	Diseases of the eye and its adnexa
From h60 to h95	Diseases of the ear and mastoid process

Table 1: Example of five clusters defined for CodiEsp Diagnostics.

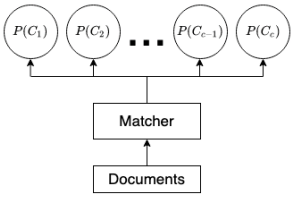

Figure 2: Overview of the Matcher module. $P(C_i)$: probability of document having a code in cluster i.

Conquer (D&C) model since dividing the original task into two simpler text classification subtasks allows us to improve the results considerably.

3.1 Clusters

As a preliminary step before training our model, we create partitions of semantically related codes based on the ontologies hierarchy. We will refer to these groups as clusters. In Table 1, we show an example of clusters defined in the CodiEsp Diagnostics corpus. Here, we used the first three letters of a code that, in the ontology, are related to a disease category.

For Codiesp-D, we created 21 clusters; for Codiesp-P 17 clusters; and for CANTEMIST 51 clusters. The clusters were defined using the categories systematized by the ontology's creators leveraging extensive work from clinicians worldwide to group semantically related codes, which gives us confidence about the quality of the selected clusters.

3.2 Matcher

As shown in Figure 2, the Matcher module assigns the probability of each document belonging to each cluster. Each document is categorized with the clusters mapped to the document labels, where each label belongs to a single cluster. This task can be formulated as multi-label text classification. Notably, the number of clusters is significantly lower than the number of labels on the corpus. For example, in the Codiesp-D subtask, the amount of different labels is 2.557, and the number of clusters is 21. This simplifies the task charged to the Matcher, classifying in fewer classes using significantly more documents per class.

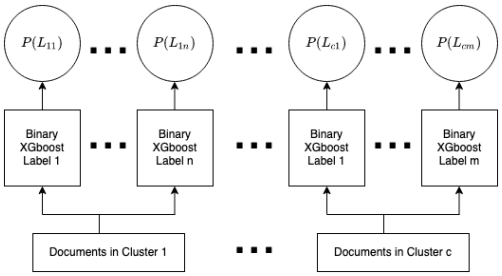

Figure 3: Overview of the Ranker module. $P(L_{ij})$: probability of document having a mention of code i in cluster j.

To perform this classification, we decided to fine-tune a transformer-based architecture, as these models have boosted the performance of NLP architectures in several NLP tasks, including text classification. Transformers models are based entirely on attention, replacing the recurrent layers most commonly used in encoder-decoder architectures with multi-headed self-attention (Vaswani et al., 2017). This aims to draw global dependencies between input and output without the need for sequential computation of Recurrent Neural Networks (RNN) (Rumelhart et al., 1986) or Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997).

3.3 Ranker

The Ranker module calculates, for each possible code, the probability of belonging to the document. This process is carried out by training a single binary model per code, following a one-vs-rest approach. Each model is trained only with documents with codes belonging to the cluster, which allows for a fine-grained differentiation between similar codes. This way, the gold labels of this task are the codes in the document.

Since each document can contain many codes, this problem, like the Matcher, can be formulated as a multi-label text classification task. However, this subtask is considered extreme since possible codes are much larger than the number of possible clusters in the other task. The Ranker module is based on the one-vs-rest approach, where the input documents are binary classifiers encoded using the TF-IDF method, and the output is fed into an Extreme Gradient Boosting (XGBoost) model (Chen et al., 2015).

We decided to use the Gradient Boosting Trees algorithm considering the computational cost of

training one model per label and also the quality of the model's predictions. Although, as previously discussed, neural networks are the go-to choice when solving an NLP task, it is not feasible to train one neural network (specifically a Transformer or LSTM) per label due to the computational costs of training in an extreme multi-label environment. Because each cluster has fewer examples than the entire corpus, even training one neural network model per cluster yields worse results because of the data scarcity issue.

3.4 Combining output of Matcher and Ranker

Having trained both the Matcher and the Ranker, the issue of how to combine the results is left. To handle this task, we implemented two approaches; one that outputs probabilities of all labels and another that predicts the labels of the document.

First, to get the probabilities of all labels, it is important to note that the output probabilities of the Ranker are not precisely the probabilities of the label because it was trained only with documents in the cluster. More rigorously, these values can be defined as the conditional probabilities of the label given that it belongs to the cluster. Therefore, to compute the probabilities of the label, we can use the Bayes Theorem,

$$P(L) = P_{Matcher}(C) * P_{Ranker}(L|C), \quad L \in C. \tag{1}$$

Where $P_{Matcher}(C)$ is the probability that the Matcher module assigns a document to cluster C, and $P_{Ranker}(L|C)$ is the conditional probability that the Ranker module assigns a label L to a document given that it belongs to cluster C.

3.5 Ensemble

Using an ensemble of strong learners only leverages different runs of the same computationally expensive training process and thus confounds the advances obtained by creating better architectures. However, since most of the previous work proposed ensemble models, we implemented an ensemble strategy to perform a fair comparison with that systems.

The ranking of all the labels is done by summing the probabilities of the ensembled models for each label, where the prediction of the final labels is a union of the predicted labels in all the ensembled models.

3.6 Data Augmentation

In addition, to improve the performance of our models, we implemented four data augmentation techniques. Three methods are based on entity mentions associated with the codes, while the other uses code descriptions. Specifically, we added new documents as follows:

- **NE Mentions**: Each entity mentioned is a new document.
- **NE Sentences**: Each sentence in the original corpus is considered as a new document.
- **NE Stripped**: New documents only with words that participate in some entity mention.
- **Definition codes**: Each definition of a code is a new document.

The first three techniques need to have a corpus in which the different labels are associated with a span of the document (Named Entity), which is widespread because most corpora are created to solve Named Entity Recognition tasks and Text Classification tasks. Not all of these data augmentation techniques are used by the Matcher and the Ranker. In fact, the Matcher uses a transformer architecture that is trained using sentences and needs semantic context, so for the Matcher, NE Stripped would only make the results worse and is not used.

4 Experiments

4.1 Datasets

We conducted our experiments with three clinical coding corpora in Spanish. Table 2 shows a summary of descriptive statistics for each corpus: CodiEsp-D, CodiEsp-p, and CANTEMIST.

- **CodiEsp**[1] (Miranda-Escalada et al., 2020b): Corpus composed of 1,000 clinical cases manually annotated using the guidelines ICD10-Clinical Modification and ICD10-Procedure. This dataset was used for two shared tasks: CodiEsp Diagnostics (CodiEsp-D) and CodiEsp Procedures (CodiEsp-P).
- **CANTEMIST**[2] (Miranda-Escalada et al., 2020a): Corpus composed of 1,301 oncologic clinical case reports annotated using the ICD-O-3 codes.

[1] https://zenodo.org/record/3837305
[2] https://zenodo.org/record/3978041

	CodiEsp-D			CodiEsp-P			CANTEMIST		
	Train	Dev	Test	Train	Dev	Test	Train	Dev	Test
Documents	500	250	250	500	250	250	501	500	300
Avg document length	410	411	414	410	411	414	894	804	812
Avg codes per document	14.4	13.7	14.7	3.9	4.2	4.4	12.8	12	12.1
Avg clusters per document	4.9	4.9	4.8	1.9	2.0	2.0	2.8	2.8	2.8
Number of different codes	2557			870			850		
Number of clusters	21			17			51		

Table 2: Descriptive statistics of the datasets.

4.2 Settings

For ease of reading, we explain the different hyperparameters and strategies used for the Matcher and the Ranker training. We used the same data splits as in previous work (Miranda-Escalada et al., 2020b,a) to guarantee a fair comparison.

Regarding the Matcher module, we fine-tuned a Biomedical version of RoBERTa in Spanish, leaving only the last layer trainable. We trained our model during 15 epochs using the Adam with weight decay optimizer (Loshchilov and Hutter, 2017), which is an improved version of Adam (Kingma and Ba, 2014) using a batch size of 25 documents. To handle overfitting, we employed a linear scheduler with warmup, which linearly increases the learning from 0 to the max learning rate during warmup and then decreased the learning rate to 0. This module was implemented using the Flair framework (Akbik et al., 2019).

To choose the optimizer, we used the defaults of the Flair framework. The number of epochs was chosen after training on the train split and evaluating on the Codiesp-D validation split. The loss reduction stagnated at epoch 10. Given that we used a Linear Scheduler that decreases the learning rate for each epoch, we used 15 epochs to ensure that we reached the best performance.

Each one-vs-rest model of the Ranker was created using the XGBoost implementation provided by the Sklearn library (Pedregosa et al., 2011). Regarding the hyperparameters, we used the exact tree method, the ratio of the negative class to the positive class as the scaling weight, the Dart enhancer, and 60% of subsample and column subsample.

To ensure the reproducibility of our results, we released an open-source library[3] with the code of our experiments. This framework allows extending the model to other datasets by simply implementing the preprocessing data functions. Likewise, data augmentation techniques can be extended to other corpora by implementing a preprocessing function that obtains the span, the document, and the mention of a code. All the experiments were performed using a GPU Nvidia DGX A100.

[3]https://github.com/plncmm/dac-divide-and-conquer

4.3 Metrics

To evaluate the performance of our model, we compute the metrics used in previous work on CodiEsp and CANTEMIST. First, we calculate the MAP, which is a widely used evaluation score for ranking problems (Miranda-Escalada et al., 2020b) and has shown good discrimination and stability (Schütze et al., 2008). MAP is defined as the mean of the average precision (AP) of all documents:

$$AP = \frac{\sum P(k) * rel(k)}{\text{number of relevant labels}},$$

where $P(k)$ is the precision at position k, and $rel(k)$ is an indicator function equaling 1 if the item at rank k is a relevant label, zero otherwise. Second, we calculated the micro average F_1 score, corresponding to the harmonic mean of the precision and recall.

These metrics were evaluated on the test set provided by the shared tasks, so comparability to other models is assured. To correctly determine whether the differences between the performance of our model and the other models are reliable or due to statistical chance, we have done five different evaluation rounds, each with a different seed, ensuring different results. The results reported are the mean of these five evaluation rounds, and the standard deviation is also reported.

Regarding the performance of the ensemble models, the report of different evaluation rounds is unfeasible due to the high computational time cost. However, the ensemble interiorizes the statistical chance because it uses 15 different instances of the architecture.

To provide a more comprehensive analysis of the architecture, we computed metrics for each one of the modules. These metrics help us gain insights into which part of the architecture levels are accept-

able and allow us to know when high scores for the architecture as a whole can be expected.

Regarding the Matcher module, we report the MAP and the F_1 score when the gold labels are the clusters. In the case of the Ranker module, we had to approach the issue of creating metrics that could evaluate its performance independently from the Matcher step, which is not straightforward. To overcome this issue, we have defined a weighted version of the metrics in which, for each cluster, we calculate the MAP and the F_1 score for that cluster's sentences. Then, the cluster metrics are aggregated and weighted by the number of sentences in each cluster. This metric indicates how well the Ranker is labeling the documents. This metric can be interpreted as what the metric would be if the Matcher had a perfect performance and thus acts as a ceiling for the DAC model's final performance.

5 Results

Table 3 shows the overall results of our model. We reported two different results for the DAC architecture: the average of five different evaluation rounds using the original approach and other results from a version that ensembled 15 different model instances.

Our base model achieves state-of-the-art results in both CodiEsp tasks, surpassing the best base model (Clinical Coding Transformers - Best (López-García et al., 2021)) by 8% in CodiEsp diagnostics and by 6% in CodiEsp procedures. Even in comparison with an ensemble of strong learners, which obtains a similar performance (Clinical Coding Transformers - Ensemble (López-García et al., 2021)), our base model surpasses their results by a small margin of 0.5% in CodiEsp Diagnostics and 0.2% in CodiEsp Procedures. Their results correspond to 15 different runs of 3 strong learners, where each language model was trained with a private oncology corpus. Unlike the mentioned work, we used only publicly available resources and a simpler architecture regarding computational cost.

Most notably, our ensemble-based version of 15 different instances of our model outperformed previous results in the CodiEsp tasks by a wide margin, outperforming state-of-the-art methods, including ensembles of strong learners, in CodiEsp-D and CodiEsp-P by 3.0% and 3.3%, respectively.

We hypothesize that the high performance of our model is explained since the original text classification task is reduced to two subtasks, where the number of possible labels is smaller. First, the Matcher module performs a text classification in which the number of labels equals the number of clusters. Second, the Ranker is trained only with documents belonging to a cluster, which allows for a fine-grained differentiation between similar codes.

We believe that the incapability to obtain state-of-the-art for CANTEMIST is because it is a simpler task than Codiesp-D and Codiesp-P. This is noticeable by looking at the performance of every model in each task. Our architecture is built to thrive under challenging tasks where a straightforward fine-tuning of a transformer is not the best approach. Nonetheless, our architecture is the third best evaluated for CANTEMIST, considering that ICB-UMA (López-García et al., 2020) and Clinical Coding Transformers (López-García et al., 2021) are from the same authors and used the same approach. Therefore, we obtained competitive results compared to state-of-the-art models and surpassed the performance of most of the systems (Miranda-Escalada et al., 2020a).

6 Module Analysis

In Table 4, we report the mean results using different language models and compare the performance with the ensemble for each corpus. We performed experiments for 15 instances of the architecture with different seeds, five using BioClinical RoBERTa, five using BioMedical RoBERTa, and five using BETO.

Notably, the MAP and F_1 scores for the Matcher are high in all experiments. This is required for the architecture to be competitive; otherwise, the error propagation leads to a low-quality final model. Another interesting finding is that we can see no significant difference between the domain-specific language models (BioMedical RoBERTa and BioClinical) across all our experiments. However, the general-domain language model we have tested (BETO) has significantly lower performance on all tasks. Finally, it is worth mentioning that the ensemble-based architecture significantly outperforms all base models at hand, at least in the MAP metric. According to the F_1 metric, it surpasses the models in the CodiEsp tasks and fails in the Cantemist corpus. This adds room for improvement in how the class prediction is combined to calculate

Model	CodiEsp-D		CodiEsp-P		CANTEMIST	
	MAP	F_1	MAP	F_1	MAP	F_1
IXA-AAA (Blanco et al., 2020)	0.593	0.009	0.425	0.008	-	-
IAM (Cossin and Jouhet, 2020)	0.521	0.687	0.493	0.522	-	-
FLE (García-Santa et al., 2020)	0.519	0.679	0.443	0.514	-	-
The Mental Strokers (Costa et al., 2020)	0.517	0.591	0.445	0.488	-	-
Vicomtech (García-Pablos et al., 2020)	-	-	-	-	0.847	**0.855**
ICB-UMA (López-García et al., 2020)	0.482	0.009	-	-	0.847	0.013
Clinical Transformers - Best (López-García et al., 2021)	0.616	-	0.514	-	0.862	-
Clinical Transformers - Ensemble (López-García et al., 2021)	0.662	-	0.544	-	**0.884**	-
Divide and Conquer (DAC)	**0.665**	**0.746**	**0.545**	**0.553**	0.788	0.712
Divide and Conquer - Ensemble (DAC-E)	**0.682**	**0.744**	**0.562**	**0.560**	0.804	0.695

Table 3: Overall results on three clinical coding datasets. Results of the Clinical Transformers are taken from the author's paper, all the other results are obtained from the competitions overview. Some results are missing because those approaches were not implemented for the corresponding tasks.

Codiesp-D	Matcher		Ranker		DaC	
	MAP	F_1	MAP	F_1	MAP	F_1
BioClinical RoBERTa - Mean	0.930	0.852	0.729	**0.726**	**0.665**	0.727
BioMedical RoBERTa - Mean	**0.938**	**0.865**	**0.730**	**0.726**	**0.665**	**0.729**
BETO - Mean	0.916	0.824	0.728	0.728	0.653	0.713
Ensemble	-	-	-	-	**0.682**	**0.744**
Codiesp-P	Matcher		Ranker		DaC	
	MAP	F_1	MAP	F_1	MAP	F_1
BioClinical RoBERTa - Mean	0.941	**0.879**	0.614	0.584	0.545	**0.536**
BioMedical RoBERTa - Mean	**0.947**	0.867	**0.617**	**0.587**	**0.546**	0.531
BETO - Mean	0.936	0.853	0.612	**0.587**	0.533	0.525
Ensemble	-	-	-	-	**0.562**	**0.560**
CANTEMIST	Matcher		Ranker		DaC	
	MAP	F_1	MAP	F_1	MAP	F_1
BioClinical RoBERTa - Mean	**0.953**	**0.900**	0.821	0.711	**0.788**	0.706
BioMedical RoBERTa - Mean	0.948	0.898	0.819	**0.713**	0.784	**0.708**
BETO - Mean	0.915	0.857	**0.822**	0.712	0.763	0.692
Ensemble	-	-	-	-	**0.804**	0.695

Table 4: Report of metrics for each module and model trained in CodiEsp-D, CodiEsp-P, and CANTEMIST. The F_1 scores of both the DaC model and the Ranker use only the first three characters of the code as the label in Codiesp-D and the first four characters of the code in Codiesp-P. We used only the first characters following the procedures of evaluating the models created by the competition. The bolded results indicate the best metric score for each module, and the underline marks the worst performance.

the F_1 metric.

7 Conclusions and Future Work

This paper proposes a novel model for clinical coding in Spanish, outperforming previous results in two datasets; CodiEsp-D and CodiEsp-P. Our method uses a Divide and Conquer approach that creates semantic groups of codes to build an architecture composed of two specialized modules: the Matcher and the Ranker.

The clinical coding task is separated into two simpler tasks solved with the modules mentioned above. First, the Matcher predicts the clusters of each document, and then the Ranker predicts the codes of each document given a cluster. This division allows us to use state-of-the-art transformers to solve the task of cluster prediction and permits a fine-grained differentiation between similar codes in a cluster using XGBoost.

Although our base model achieves better results than previous ensemble-based models, we included the results of an ensemble strategy to perform a fair comparison with previous work. Our experimental results demonstrate that ensembling models yield better results than our base model. Furthermore, our DaC approach allows us to identify where future research can have a greater impact on improving accuracy. The results of each module indicate that there is more potential for improvement focusing on the Ranker module.

Future directions include implementing and test-

ing the Divide and Conquer model on other multi-label text classification corpora. First, we expect to test the DaC architecture on clinical corpora in other languages, including languages with more resources, such as English. Second, we expect to test the architecture on other extreme multi-label classification corpora. This poses a challenge since the number of labels we have processed thus far, although very vast, falls into the category of small extreme multi-label classification datasets (Bhatia et al., 2016). We expect to encounter issues with the training time required to process other large corpora, forcing us to modify the library to optimize the speed.

In terms of improving the performance using this architecture, we identify opportunities to optimize the number of layers that we left fine-tuneable in the Matcher module, given that we have seen research that shows that fine-tuning more layers provides better results (Lee et al., 2019). Also, for the Ranker, we know that XGBoost can be trained with a ranking objective function, thus providing an alternative to the one-vs-rest approach. Implementing the Ranker using this approach would be faster to train and may provide similar or better results. In addition, we would like to improve data augmentation techniques by improving NER models. This can be achieved by using contextualized embeddings at the character level, which has been shown to improve the performance of models on various NLP tasks (Rojas et al., 2022a,b).

Finally, the DaC architecture is a black box when defining which labels to assign for each document. Recently, explainability features of the different architectures are gaining more relevance. It is paramount that the model's predictions are understood to help the user make appropriate choices (Duque et al., 2021). We expect to develop explainability to the labels predicted by providing textual queues of what features the model used to choose each label. The textual queues that support label assignment can be provided by the Ranker leveraging the explainability features of tree ensembles (Petkovic et al., 2018), and the textual queues that support the cluster choice can be obtained using the attention weights of the transformer model (Liu et al., 2021).

Limitations

Although our approach achieved excellent results across all the corpora in this research, they have clear limitations. The main drawback is that to apply this approach, it is necessary to have codes that can be clusterized. In fact, only a thorough categorization of similar codes into groups yields accurate results. Another major drawback is that the architecture predicts codes at the document level, thus having information that is not as complete as an entity-level prediction.

Finally, one limitation of the Matcher module is that it has a maximum document size of 512 tokens since it uses pre-trained transformers, which can contribute to losing important information on the cluster prediction process.

Acknowledgements

This work was funded by ANID Chile: Basal Funds for Center of Excellence FB210005 (CMM) and FB210017 (CENIA); Millennium Science Initiative Program ICN17_002 (IMFD) and ICN2021_004 (iHealth), and Fondecyt grant 11201250. Regarding hardware, the research was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02) and the Patagón supercomputer of Universidad Austral de Chile (FONDEQUIP EQM180042).

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Kush Bhatia, Kunal Dahiya, Himanshu Jain, Purushottam Kar, Anshul Mittal, Yashoteja Prabhu, and Manik Varma. 2016. The extreme classification repository: Multi-label datasets and code.

Alberto Blanco, Alicia Pérez, and Arantza Casillas. 2020. IXA-AAA at CLEF eHealth 2020 CodiEsp. Automatic Classification of Medical Records with Multi-label Classifiers and Similarity Match Coders. In *CLEF (Working Notes)*.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S Dhillon. 2020. Taming pre-trained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3163–3171.

Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4):1–4.

Sébastien Cossin and Vianney Jouhet. 2020. IAM at CLEF eHealth 2020: Concept Annotation in Spanish Electronic Health Records. In *CLEF (Working Notes)*.

Joao Costa, Inês Lopes, André V Carreiro, David Ribeiro, and Carlos Soares. 2020. Fraunhofer aicos at clef ehealth 2020 task 1: Clinical code extraction from textual data using fine-tuned bert models. In *CLEF (Working Notes)*.

Andres Duque, Hermenegildo Fabregat, Lourdes Araujo, and Juan Martinez-Romo. 2021. A keyphrase-based approach for interpretable ICD-10 code classification of Spanish medical reports. *Artificial Intelligence in Medicine*, 121:102177.

Aitor García-Pablos, Naiara Perez, and Montse Cuadros. 2020. Vicomtech at cantemist 2020. In *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020), CEUR Workshop Proceedings*.

Nuria García-Santa, Kendrick Cetina, L Cappellato, C Eickhoff, N Ferro, and A Nevéol. 2020. FLE at CLEF eHealth 2020: Text Mining and Semantic Knowledge for Automated Clinical Encoding. In *CLEF (Working Notes)*.

Ira Goldstein, Anna Arzumtsyan, and Özlem Uzuner. 2007. Three approaches to automatic assignment of ICD-9-CM codes to radiology reports. In *AMIA Annual Symposium Proceedings*, volume 2007, page 279. American Medical Informatics Association.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Rajvir Kaur, Jeewani Anupama Ginige, and Oliver Obst. 2021. A systematic literature review of automated ICD coding and classification systems using discharge summaries. *arXiv preprint arXiv:2107.10652*.

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

Leah S Larkey and W Bruce Croft. 1995. Automatic assignment of ICD-9 codes to discharge summaries. Technical report, Technical report, University of Massachusetts at Amherst, Amherst, MA.

Jaejun Lee, Raphael Tang, and Jimmy Lin. 2019. What would elsa do? freezing layers during transformer fine-tuning. *arXiv preprint arXiv:1911.03090*.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 115–124, New York, NY, USA. Association for Computing Machinery.

Shengzhong Liu, Franck Le, Supriyo Chakraborty, and Tarek Abdelzaher. 2021. On exploring attention-based explanation for transformer models in text classification. In *2021 IEEE International Conference on Big Data*, pages 1193–1203. IEEE.

Guillermo López-García, José María Jerez, and Francisco José Veredas. 2020. ICB-UMA at CANTEMIST 2020: Automatic ICD-O Coding in Spanish with BERT. In *IberLEF@ SEPLN*, pages 468–476.

Guillermo López-Garcıa, José María Jerez, and Francisco José Veredas. 2020. ICB-UMA at CLEF e-health 2020 Task 1: Automatic ICD-10 coding in Spanish with BERT. In *Proc. Work. Notes CLEF, Conf. Labs Eval. Forum, CEUR Workshop*, pages 1–15.

Guillermo López-García, José María Jerez, Nuria Ribelles, Emilio Alba, and Francisco J Veredas. 2021. Transformers for clinical coding in spanish. *IEEE Access*, 9:72387–72397.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Antonio Miranda-Escalada, Eulalia Farré, and Martin Krallinger. 2020a. Named entity recognition, concept normalization and clinical coding: Overview of the cantemist track for cancer text mining in spanish, corpus, guidelines, methods and results. In *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020), CEUR Workshop Proceedings*.

Antonio Miranda-Escalada, Aitor Gonzalez-Agirre, Jordi Armengol-Estapé, and Martin Krallinger. 2020b. Overview of automatic clinical coding: Annotations, guidelines, and solutions for non-english clinical cases at codiesp track of clef ehealth 2020. In *CLEF (Working Notes)*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Dragutin Petkovic, Russ Altman, Mike Wong, and Arthur Vigil. 2018. Improving the explainability of random forest classifier–user centered approach. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 204–215. World Scientific.

Matias Rojas, Felipe Bravo-Marquez, and Jocelyn Dunstan. 2022a. Simple yet powerful: An overlooked architecture for nested named entity recognition. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2108–2117, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Matías Rojas, Jocelyn Dunstan, and Fabián Villena. 2022b. Clinical flair: A pre-trained language model for Spanish clinical natural language processing. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 87–92, Seattle, WA. Association for Computational Linguistics.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to Information Retrieval*, volume 39. Cambridge University Press.

Fei Teng, Yiming Liu, Tianrui Li, Yi Zhang, Shuangqing Li, and Yue Zhao. 2022. A review on deep neural networks for icd coding. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.