

# Vicomtech at MESINESP2: BERT-based Multi-label Classification Models for Biomedical Text Indexing

Aitor García-Pablos, Naiara Perez and Montse Cuadros

*SNLT group at Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), Mikeletegi Pasealekua 57, Donostia/San-Sebastián, 20009, Spain*

## Abstract

This paper describes the participation of the Vicomtech NLP team in the MESINESP2 shared task. The challenge consists in the development of systems for the automatic indexing with DeCS codes of health-related documents in Spanish. The systems submitted by Vicomtech are multilabel classifiers based on pre-trained BERT models. We have experimented with multiple ways of representing the documents, such as encoding DeCS term glosses along with the input text. According to the official evaluation results, our systems are surpassed by other competing teams—despite being fast and achieving good precision, we fall behind especially in recall metrics. Overall, the task remains challenging even for the best performing systems and there is ample room to advance the state of the art for this particular task.

## Keywords

Biomedical Text, Automatic Indexing, DeCS, Spanish

## 1. Introduction

The MESINESP2 shared task [1], similar to the first MESINESP edition [2], is an open BioASQ [3] competition to develop automatic systems for the semantic indexing of Spanish documents with DeCS<sup>1</sup>, a structured medical vocabulary derived from the Medical Subject Headings (MeSH) [4]. DeCS comprises 34,294 descriptors and qualifiers.

The shared task is divided into three subtracks, each targeted to a different type of health-related document: scientific literature, clinical trials and patents, respectively. We have participated in all the subtracks implementing variations of a Transformers-based [5] multi-label classification model. In particular, our models feature a pre-trained BERT model [6] to encode the input text and, in some versions, inject external knowledge (i.e. DeCS term glosses) to the model. Despite our scores lagging behind the best competing systems, our team ranks third in Subtrack 1 and second in Subtrack 2. Still, the overall challenge results show that there is room for improvement and future work.


The rest of the document is structured as follows. Section 2 introduces the data provided by the organizers of the challenge, with a special focus on the DeCS code imbalance and how we tackle this problem. Sections 3 and 4 describe our submitted systems and the training setup,


---

*CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania*

✉ [agarcia@vicomtech.org](mailto:agarcia@vicomtech.org) (A. García-Pablos); [nperez@vicomtech.org](mailto:nperez@vicomtech.org) (N. Perez); [mcuadros@vicomtech.org](mailto:mcuadros@vicomtech.org) (M. Cuadros)

🆔 0000-0001-9882-7521 (A. García-Pablos); 0000-0001-8648-0428 (N. Perez); 0000-0002-3620-1053 (M. Cuadros)

 © 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

<sup>1</sup><http://decs.bvsalud.org/I/homepagei.htm>

**Table 1**

Size of the shared task corpus in terms of number of documents per subtrack and split

	<b>Training</b>	<b>Development</b>	<b>Background</b>	<b>Testing</b>
<b>Subtrack 1</b> <i>Scientific literature</i>	237,574	1,065	10,174	500
<b>Subtrack 2</b> <i>Clinical trials</i>	3,560	147	8,919	250
<b>Subtrack 3</b> <i>Patents</i>	0	115	6,000	150

respectively. Section 5 presents the official results. In Section 6, we discuss some decisions taken during the development and training phases, inherent flaws of our systems, and potential improvements. Finally, Section 7 provides some concluding remarks and future work hints.

## 2. Data description

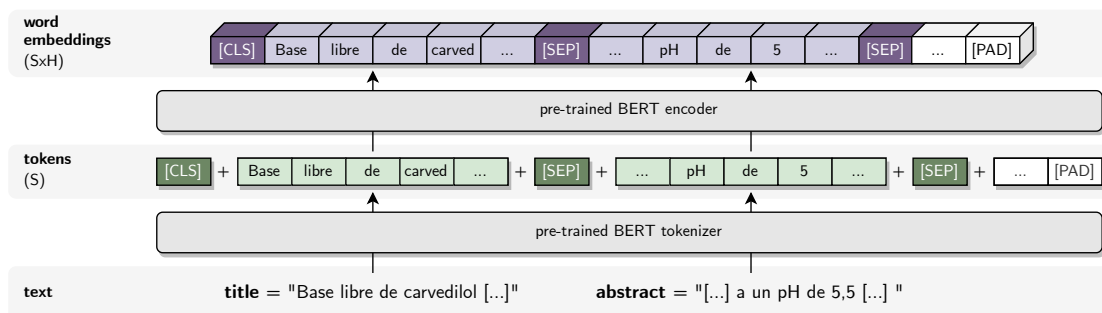
MESINESP2 is organized into three subtracks, each focused on health-related documents of a different genre [7]. The sizes of the datasets per subtrack and split are shown in Table 1. The information available for each document of the corpus is the following:

- Document ID: a unique identifier for the document.
- Title: the title of the document.
- Abstract: the abstract of the document, which is the main source of text for the task.
- Metadata: journal, year and database.
- DeCS codes: the DeCS codes that characterize the content of the document.

The objective of the MESINESP competition is to develop a system capable of predicting the correct set of DeCS codes for any new document. From the total of 34,294 codes of the DeCS terminology, only around 22,000 are present in the training data. Furthermore, the frequency in which the codes occur is highly unbalanced: a minority of codes occur in more than 80% of the training documents, while the majority of codes are way more sparse, with less than a hundred examples in the whole dataset. This problem is exacerbated by the fact that we use a multi-label classification approach, so the codes cannot be easily balanced.

A naive sub-sampling or over-sampling approach would sample the codes grouped by documents, and the imbalance would persist. Further, the codes that barely appear in a few tens of documents in the whole corpus are very unlikely to be learnt by the model, due to the lack of representation. We have addressed this problem by applying a minimum support cut-off. That is, the codes with a frequency lower than a certain preset value are not taken into account for training. A large minimum support cut-off would lead to discard too many codes, while too small a cut-off value would keep many underrepresented labels in the output vocabulary.

In order to maintain an equilibrium between these two extremes, we have calculated a cut-off value that minimizes the number of codes in the resulting vocabulary while keeping as many



**Figure 1:** Representation of a MESINESP document title and abstract using a BERT model.

codes as possible from those that occur in the development set. The resulting cut-off value calculated thus over the Subtrack 1 dataset is 80. That is, we have ignored all the codes that appear fewer than 80 times in the training set documents. This reduces the size of the output vocabulary from more than 22,000 codes to 3,274, which still represent 82% of the codes present in the development set.

### 3. System description

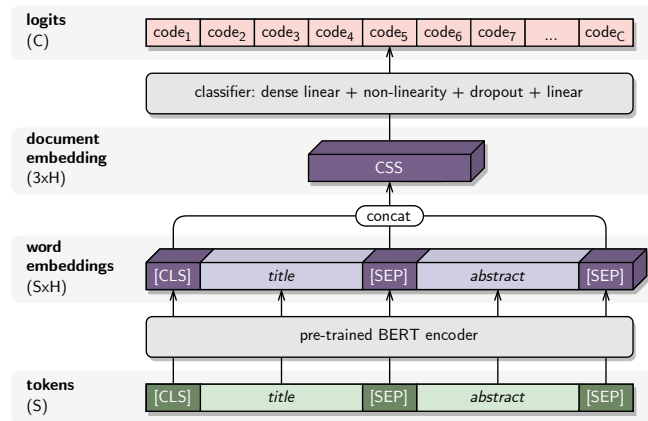
#### 3.1. Input representation

We use the titles and abstracts of the documents to be indexed as the main sources of information. In order to feed these fields to the BERT models that will generate the contextual word embeddings, we concatenate them using the usual BERT representation for two texts: the special [CLS] token, followed by the tokenised title, the special [SEP] token, the tokenised abstract, and a second [SEP] token (see Figure 1).

BERT-base models have a hard limit of 512 tokens, including special characters. The average length of the abstracts in the training set after tokenisation with the corresponding BERT tokeniser is around 300 tokens, with a standard deviation of about 120 tokens. That is, a large percentage of the documents fit in the model. The few ones that do not are simply truncated. We assume that even in those cases in which the last words of the abstract are omitted, the amount of information encoded in the first few hundreds of words is enough to predict the most salient DeCS codes for a given document. This assumption is supported by the fact that, during some preliminary experiments, we did not observe major differences in the development set scores when varying the maximum allowed document length between 300 and 500 BERT tokens.

#### 3.2. Architectures

We have experimented with two different architectures. We henceforth refer to these systems as CSS and LABELGLOSSES. Both systems are multi-label classifiers built on top of a Transformers model. Given a document, they can predict any number of labels, from 0 to  $C$ ,  $C$  being the size



**Figure 2:** Architecture of the CSS model for multi-label classification of the documents.

of the output vocabulary, i.e., 3,274 (in practice, each document is associated to a small number of labels—usually less than 10). The difference between the architectures lies in how they use the contextual word embeddings obtained from the Transformer model.

Figure 2 shows a diagram of the CSS model. The token contextual-embeddings obtained from the BERT model need to be gathered or processed in a way that provides a fixed length representation, usually called document embedding, that serves as input to a classification head. There are different ways to obtain such a representation.

The most direct and straightforward approach is to use the special [CLS] token to act as the document summary. For a model built on a BERT-base architecture, the [CLS] token is a vector of 768 values. After some experimentation, the use of just this token led to poor results. Our hypothesis is that summarizing the whole document into 768 values aiming at discriminating several thousands of possible classes (i.e. DeCS codes) leads to a choke point. That is, the information represented in the [CLS] token alone is too compressed to serve as the input for a classifier with such a high number of output classes.

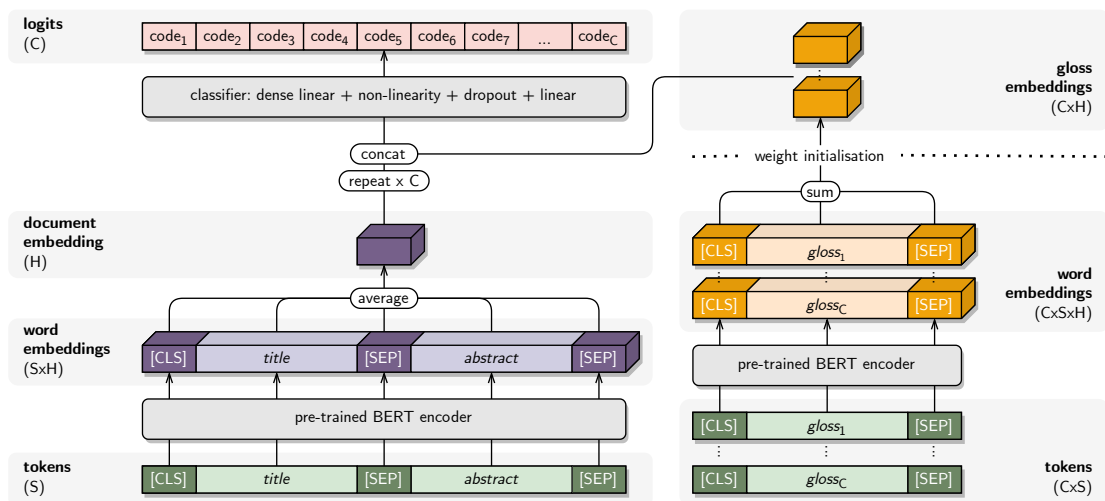
To overcome this problem, we draw on other tokens that are always present in every document due to the way we represent them: the [SEP] tokens. For each document, we concatenate the [CLS] token and the two [SEP] tokens into a single vector of  $3 \times 768$  values (hence the name of the model). This larger document embedding is then used as input for the classification head.

The classification head is composed of a dense layer, followed by a nonlinear function, a dropout layer and a linear layer that maps the document-embedding size into a label space.

### 3.2.1. LABELGLOSSES model

The initial components of the LABELGLOSSES model are very similar to the CSS model, the difference between the two architectures being that the LABELGLOSSES model contains an encoded representation of the glosses that describe the DeCS codes (see Figure 3). Table 2 shows some examples of such glosses.

Prior to the training phase, the DeCS glosses are encoded using the same pre-trained BERT



**Figure 3:** Architecture of LABELGLOSSES model. DeCS codes’ glosses are encoded into embeddings and paired to each document-embedding as the input for the classification head.

**Table 2**

Examples of glosses for several DeCS terms, together with their code and name

Code	Term	Gloss
D003970	Diastema	Espacio entre dos dientes adyacentes en el mismo arco dental. (Dorland, 27th ed)
D007962	Leucocitos	Células sanguíneas blancas. Estas incluyen a los leucocitos granulares (BASOFILOS, EOSINOFILOS y NEUTROFILOS) así como a los leucocitos no granulares (LINFOCITOS y MONOCITOS).
DDCS034870	Mareógrafo	Instrumento para registrar y medir las oscilaciones de las mareas. (Material IV - Glosario de Protección Civil, OPS, 1992)
D011203	Pobreza	Acción y efecto de empobrecer o empobrecerse. (Fuente: Diccionario de la lengua española. Real Academia Española. Disponible en: <a href="https://dle.rae.es/?id=ErpRftz">https://dle.rae.es/?id=ErpRftz</a> )

model that will be used for training. First, we strip all parenthetical content from the glosses, because such content is often a citation or other irrelevant boilerplate. The gloss encoding process consists in summing the contextual embeddings obtained from the tokens of each gloss, ignoring padding positions. The resulting vectors are used to initialize the DeCS gloss embeddings layer inside the model. This layer is of size  $C \times H$ ,  $C$  being the number of codes in the output vocabulary and  $H$  the size of BERT embeddings (i.e. 768 for a BERT-base architecture). These embeddings are fine-tuned during training.

During training, the document embedding is obtained by averaging the contextual token embeddings from the BERT model. Then, this document embedding is combined with every gloss embedding, forming pairs:  $(document, gloss_1), (document, gloss_2), \dots, (document, gloss_C)$ . Each pair consists of two vectors of size  $H$  that are concatenated to obtain a single vector of

size  $H \times 2$ . This combined vector is the input to a classification head.

The classification head of the LABELGLOSSES model is the same as the CSS model: a dense layer followed by a nonlinear function, a dropout layer and a linear layer.

### 3.3. Output handling

The output of the model (be it CSS or LABELGLOSSES) is an individual score ranging between 0 and 1 for each DeCS code in the output vocabulary. When the model is confident about predicting a certain code, its corresponding score gets closer to 1, and vice versa. A threshold needs to be chosen to decide when a given score must be interpreted as the model predicting the corresponding code for the given input.

With a threshold of 0, the model would predict all the codes regardless of the input, maximizing the recall but minimizing the precision. A threshold of 1 would mean that the model would never predict any code at all. In the absence of further information, a threshold of 0.5 is a reasonable default, but could be suboptimal.

In this work, we have used the development sets provided for each subtrack to find out the decision threshold that better balances the precision and recall, thus achieving the best possible F1-score for each trained model. The actual thresholds are reported in the next section.

## 4. Training setup and submitted systems

We have participated in all the task subtracks with several variations of the CSS and LABELGLOSSES models, listed in Table 3.

For Subtrack 1, we used IXAmBERT [8] as the pre-trained core model, a BERT-base model pre-trained for Spanish, Basque and English. The runs of Subtracks 2 and 3 use the fine-tuned model resulting from Subtrack 1 as starting point for their own fine-tuning. The reason for this is that the training dataset for Subtrack 2 is small, while there is no training data at all for Subtrack 3 (see Table 1). To measure the impact these choices might have, Subtrack 2 includes a submission that parts from IXAmBERT, and we submit to Subtrack 3 a run that has not been fine-tuned on the subtrack data.

The models have been trained using a GPU NVIDIA 2080ti of 11GB. The training run for a maximum of 200 epochs, with an early-stopping patience of 50 epochs. Under these conditions, the training of the CSS model required 3-4 days, while the training of the LABELGLOSSES model required around a week to get to the best result validated in the development set. Other training hyperparameters for the described systems are shown in Table 4.

The resulting systems process the background sets (6,000 to 10,000 documents) in 2-3 minutes at a speed of  $\sim 80$  documents per second using 1 NVIDIA RTX 1080ti GPU.

## 5. Results

Table 5 shows the official results of the competition, including the results of all our runs and the results of the winner system per subtrack. Internal evaluations with the development set showed scores around 44 micro-averaged F1-score for Subtrack 1. However, in the test set our

**Table 3**

Runs submitted to the competition, characterized by model architecture, pre-training of the encoder model, data split used for fine-tuning and inference threshold value

	Run	Architecture	Pre-train	Fine-tuned on	Threshold
<b>Subtrack 1</b>	1.1	CSS	IXAmBERT	Subtrack 1 trainset	0.25
<i>Scientific literature</i>	1.2	CSS	IXAmBERT	Subtrack 1 trainset	0.30
	1.3	CSS	IXAmBERT	Subtrack 1 trainset	0.35
	1.4	LABELGLOSSES	IXAmBERT	Subtrack 1 trainset	0.10
	1.5	LABELGLOSSES	IXAmBERT	Subtrack 1 trainset	0.20
<b>Subtrack 2</b>	2.1	CSS	IXAmBERT	Subtrack 2 trainset	0.25
<i>Clinical trials</i>	2.2	CSS	Run 1 CSS	Subtrack 2 trainset	0.20
	2.3	CSS	Run 1 CSS	Subtrack 2 trainset	0.25
	2.4	CSS	Run 1 CSS	Subtrack 2 trainset	0.30
<b>Subtrack 3</b>	3.1	CSS	Run 1 CSS	<i>none</i>	0.05
<i>Patents</i>	3.2	CSS	Run 1 CSS	Subtrack 3 devset	0.05
	3.3	CSS	Run 1 CSS	Subtrack 3 devset	0.10
	3.4	CSS	Run 1 CSS	Subtrack 3 devset	0.15
	3.5	CSS	Run 1 CSS	Subtrack 3 devset	0.20

**Table 4**

Training hyperparameters

Hyperparameter	Value	Hyperparameter	Value
Max. sequence length	300	Max training epochs	200 epochs
Batch size	16	Early stopping patience	50 epochs
Optimiser	AdamW [9]	Dropout rate	0.1
Learning rate	4E-5	Monitored metric	micro F1-score
Learning rate warm-up	linear, 2 epochs	Min. support cutoff	80
Non-linearity	Mish [10]		

best system scores 38 points, 10 points below the best competing system in Subtrack 1, and 8 points in Subtrack 2. We achieve a reasonable level of precision, only 3 points below the winning system, but recall scores fall behind. This places us in the third and second position for Subtracks 1 and 2, respectively, after the groups of systems by two other teams.

Unsurprisingly, the results in Subtrack 3 are lower than in the other two, as the runs submitted to this subtrack have seen very little to no in-domain training data, and do not exploit any other source of domain knowledge. It is noteworthy that having fine-tuned the model on just 115 examples—i.e. the development examples available for this subtrack—has had a remarkable positive impact, increasing recall by 12 points (compare Run 3.1 and 3.2).

Using the LABELGLOSSES architecture in Subtrack 3 might have helped mitigate the lack of training data, although it seems unlikely given the difference between CSS and LABELGLOSSES in Subtrack 1. Our attempts to include expert knowledge in the system by encoding DeCS glosses have not had a beneficial impact on the results. In fact, the results obtained by the LABELGLOSSES runs are slightly lower for all the metrics.

**Table 5**

Official results per subtrack and run (the subscript numbers next to the architecture names indicate the inference threshold values), including the best competing system per subtrack

	Run	F1	P	R	Acc
<b>Subtrack 1</b> <i>Scientific literature</i>	1.1 CSS <sub>0.25</sub> w/ IXAmBERT	38.23	45.09	33.18	23.99
	1.2 CSS <sub>0.30</sub> w/ IXAmBERT	38.25	46.22	32.62	24.05
	1.3 CSS <sub>0.35</sub> w/ IXAmBERT	38.01	47.10	31.86	23.90
	1.4 LABELGLOSSES <sub>0.10</sub> w/ IXAmBERT	37.04	45.26	31.34	23.13
	1.5 LABELGLOSSES <sub>0.20</sub> w/ IXAmBERT	37.46	45.60	31.79	23.23
	Best System (BERTDeCS version 4)	<b>48.37</b>	<b>50.77</b>	<b>46.18</b>	<b>32.61</b>
<b>Subtrack 2</b> <i>Clinical trials</i>	2.1 CSS <sub>0.25</sub> w/ IXAmBERT	24.85	27.21	22.87	13.84
	2.2 CSS <sub>0.20</sub> w/ Run 1 CSS	28.10	28.88	27.36	16.31
	2.3 CSS <sub>0.25</sub> w/ Run 1 CSS	28.19	29.33	27.15	16.36
	2.4 CSS <sub>0.30</sub> w/ Run 1 CSS	28.07	29.24	26.78	16.29
	Best System (BERTDeCS version 2)	<b>36.40</b>	<b>36.66</b>	<b>36.14</b>	<b>22.42</b>
<b>Subtrack 3</b> <i>Patents</i>	3.1 CSS <sub>0.05</sub> w/ Run 1 CSS (no fine-tuning)	19.68	27.00	15.48	10.76
	3.2 CSS <sub>0.05</sub> w/ Run 1 CSS	26.51	25.47	27.64	15.72
	3.3 CSS <sub>0.10</sub> w/ Run 1 CSS	28.34	31.88	25.51	16.89
	3.4 CSS <sub>0.15</sub> w/ Run 1 CSS	29.08	35.96	24.40	17.29
	3.5 CSS <sub>0.20</sub> w/ Run 1 CSS	29.21	38.90	23.28	17.25
	Best System (BERTDeCS version 2)	<b>45.14</b>	<b>44.87</b>	<b>45.41</b>	<b>30.05</b>

A final observation can be made for Runs 2.1 and 2.3, were the former’s fine-tuning starts with IXAmBERT while the latter uses the CSS model resulting from Run 1. The knowledge captured from the Subtrack 1 data has helped raise all metrics, particularly recall (+4 points).

## 6. Discussion

During the training of our systems and their variations, we have made several noteworthy observations. First, the validation scores for all the systems progressed at different paces, some faster than others, towards a plateau of around 45-50 F1-score points in the development sets. Models and hyperparameter variations made little difference. We assume that this plateau is the limit of what the proposed models can learn to generalize from the training data, in particular for the least frequent DeCS codes. For this reason, we tried to inject external knowledge to the model by encoding DeCS term glosses. The proposed approach has not helped in this regard.

Second, our systems show a clear imbalance between precision and recall. We hypothesise that the imbalance is related, among others, to the exclusion of DeCS codes from the training data when applying the minimum support cut-off value, although further research would be necessary to confirm this or to uncover interactions between other elements of the systems that might be having this effect.

One such relevant element is the decision threshold, which we use to interpret the output of our models. For each model, we have computed the global threshold that maximises the F1-score in the corresponding development set. That is, the same threshold applies to all the modeled



DeCS codes, regardless of the degree of confidence the model might have with respect to each individual code. Given the imbalance of code frequencies in the training data, the confidence is bound to vary greatly. Thus, a decision threshold better tailored to each DeCS code could benefit the precision-recall balance of the results.

It will be interesting to learn how the winning systems have addressed all these problems. For instance, the official results show that the group of systems that obtained the second position in Subtrack 1, surpassing our models, rank in third position in Subtrack 2, just behind our models. It would be interesting to study the cause for this variation and assess whether the difference lies in the approaches implemented or just in the training procedure.

Overall, the systems we have submitted, in particular the CSS model, are not complex, and the scores achieved are lower than expected given the results obtained on the development sets. However, our systems are lightweight and fast, being able to process about 80 documents per second in a commodity GPU, consuming less than 4GB of GPU memory, which enables real-time processing scenarios.

## 7. Conclusions

In these working notes we describe our participation in the MESINESP2 shared task, focused on the medical document indexing in Spanish. We have presented two systems based on Transformers, in particular using BERT-base pre-trained models to encode the text information and to perform a multi-label classification over the large DeCS codes vocabulary.

The simpler approach relies on combining special BERT-encoded tokens as input for a classification head. In this sense, it is a straightforward model that works fast. The second proposed approach shares key components, namely, the BERT-base model and the multi-label classification nature of the model. The main difference is that it adds an extra layer of DeCS code embeddings, which are meant to encode the meaning of each modelled DeCS code. The embeddings are initialized from the BERT-encoded glosses that provide human-readable definitions of the DeCS codes.

Despite our experiments on the development set having yielded scores around 44 F1-score points, our best results in the test set reach only around 38, falling 10 points behind the best competing system. Even with these lower results, our team achieves the third position among the competing teams in Subtrack 1, and the second position in Subtrack 2.

As future work, we have come across several issues that need to be addressed in order to better understand the performance of our systems and improve their results. Most interestingly, we have observed that regardless of the approach and hyperparameter variations, the models reached a similar plateau in the validation score in all our experiments. This suggests that our approaches meet their limit there, and that additional external knowledge is needed to cross it. Thus, we will focus on better and more efficient representations of the DeCS codes, for instance including their hierarchical nature. It would also be interesting to explore approaches related to semantic information retrieval.

In conclusion, the task remains challenging regardless of the model and approach, with the winning system having achieved scores lower than 50 F1-score points. Further research will be necessary to improve the state of the art of the task proposed by MESINESP2.

## Acknowledgments

This work has been partially funded by the projects DeepText (KK-2020-00088, SPRI, Basque Government) and DeepReading (RTI2018-096846-B-C21, MCIU/AEI/FEDER, UE).

## References

- [1] L. Gasco, A. Nentidis, A. Krithara, D. Estrada-Zavala, R.-T. Murasaki, E. Primo-Peña, C. Bojo-Canales, G. Paliouras, M. Krallinger, Overview of BioASQ 2021-MESINESP track. Evaluation of Advance Hierarchical Classification Techniques for Scientific Literature, Patents and Clinical Trials. (2021).
- [2] C. Rodriguez-Penagos, A. Nentidis, A. Gonzalez-Agirre, A. Asensio, J. Armengol-Estapé, A. Krithara, M. Villegas, G. Paliouras, M. Krallinger, Overview of MESINESP8, a Spanish Medical Semantic Indexing Task within BioASQ 2020, in: Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, 2020, pp. 1–12.
- [3] A. Nentidis, G. Katsimpras, E. Vandorou, A. Krithara, L. Gasco, M. Krallinger, G. Paliouras, Overview of BioASQ 2021: The ninth BioASQ challenge on Large-Scale Biomedical Semantic Indexing and Question Answering. (2021).
- [4] C. E. Lipscomb, Medical subject headings (mesh), Bulletin of the Medical Library Association 88 (2000) 265–266.
- [5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention Is All You Need, in: Proceedings of the Thirty-first Conference on Advances in Neural Information Processing Systems (NeurIPS 2017), 2017, pp. 5998–6008.
- [6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.
- [7] L. Gasco, M. Krallinger, M. Antonio, MESINESP2 Corpora: Annotated Data for Medical Semantic Indexing in Spanish, 2021. Funded by the Plan de Impulso de las Tecnologías de las del Lenguaje (Plan TL).
- [8] A. Otegi, A. Agirre, J. A. Campos, A. Soroa, E. Agirre, Conversational Question Answering in Low Resource Scenarios: A Dataset and Case Study for Basque, in: Proceedings of The 12th Language Resources and Evaluation Conference, 2020, pp. 436–442.
- [9] I. Loshchilov, F. Hutter, Decoupled Weight Decay Regularization, in: Proceedings of the Seventh International Conference on Learning Representations (ICLR 2019), 2019, pp. 1–18.
- [10] D. Misra, Mish: A Self Regularized Non-Monotonic Neural Activation Function, arXiv:1908.08681 (2019) 1–13.