# CoLe and LYS at BioASQ MESINESP Task: large scale multilabel text categorization with sparse and dense indices.

Francisco J. Ribadas-Pena[1], Shuyuan Cao[1] and Elmurod Kuriyozov[2]

[1] *Grupo COLE, Departamento de Informática, Universidade de Vigo*
*E.S. Enxeñaría Informática, Campus As Lagoas, Ourense 32004, Spain*
[2] *Grupo LYS, Departamento de Computación y Tecnologías de la Información, Universidade de A Coruña*
*Facultade de Informatica, Campus de Elviña, A Coruña 15071, Spain*

**Abstract**

In this paper we describe our participation in the second edition of MESINESP shared-task in the BioASQ biomedical semantic indexing challenge. The system employed in this participation tries to exploit different strategies for the use of similarity between documents to build a multi-label classifier that assigns DeCS descriptors to new documents from the descriptors previously assigned to similar documents. We have implemented and evaluated two complementary proposals: (1) the use of sparse document representations, based on the extraction of linguistically motivated index terms and their subsequent indexing using Apache Lucene and (2) the use of indices storing dense representations of training documents obtained by means of sentence level embeddings. The results obtained in official runs were far from the best performing systems, but we believe that our approach offers an acceptable performance taking into account the minimum processing requirements that the proposed document similarity scheme supposes.

**Keywords**

Information Retrieval, Dense Representation, Sparse Textual Representation, Multi-Label Classification



## 1. Introduction

The MESINESP2 [1] shared-task on medical semantic indexing in Spanish is part of the BioASQ [5] 2021 challenge. Content indexing using structured vocabularies is a critical task in the management of large textual collections in scientific and technical domains and it is essential to make possible sophisticated search engines to help researchers in accessing to relevant information. Although Spanish is one of the most spoken languages, most of the previous efforts and advances in semantic indexing have been oriented exclusively to English texts and the aim of the MESINESP challenge series was to evaluate the state-of-art and to promote the research in semantic indexing of Spanish scientific literature.

The second edition of MESINESP shared-task asked participant teams to label test documents with codes from DeCS (*Descriptores en Ciencias de la Salud*), a controlled hierarchical vocabulary which is a translation and extension of the MeSH (*Medical Subjects Headings*) thesaurus.

*CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania*
ribadas@uvigo.es (F. J. Ribadas-Pena); shuyuan.cao@uvigo.es (S. Cao); e.kuriyozov@udc.es (E. Kuriyozov)

This edition was composed of three sub-tracks, dealing with scientific literature (Sub-track 1, MESINESP-L), clinical trials (Sub-track 2, MESINESP-T), and biomedical patents (Sub-track3, MESINESP-P).

Our team has participated in the three sub-tracks evaluating the adequacy of various approaches based on textual similarity. The methods used in the three sub-tracks have been essentially the same and are an extension of those used in our participation at the previous edition of this challenge [8, 7]. The starting idea of our method is to identify the training documents most similar to a given test document. Using the set of descriptors assigned to these similar documents we construct the list of candidate labels to be returned as a result.

In our experiments and in the submitted runs we have evaluated different approaches for identifying this list of similar training documents. As in previous editions of the BioASQ challenge, we have used several natural language processing (NLP) techniques to extract linguistically motivated representations of the training documents that are stored in an Apache Lucene textual index. This index is later queried with the contents of each test document to retrieve the most similar documents. In addition to using this kind of sparse document representations we have proposed the use of dense representations based on sentence-level embeddings. The dense vectors extracted from train documents are indexed in order to locate, during the categorization phase, the set of vectors closest to the dense vectors extracted from the sentence-level embeddings of the test documents to be annotated. Additionally, we have tried to improve the performance of our sparse method based on Apache Lucene using an alternative type of index which is based on the creation of inverse DeCS code profiles that link index terms extracted from the documents with the DeCS tags with which they have a high co-occurrence level.

The rest of this paper is organized as follows. Section 2 describes the details of our method based on sparse representations on Apache Lucene indices. The generation of inverse DeCS codes profiles is also described in this section. Section 3 details the use of dense representations extracted from sentence-level embeddings. Section 4 provides the preliminary experiments with these methods that were used to carry out the parameterization of the official runs sent to the challenge. Finally, in section 5 we present the details of these official runs and provide a discussion of the results obtained by our approaches in the challenge.

## 2. Similarity with sparse representations

Methods following $k$ nearest neighbors ($k$-NN) approaches have been widely used in the context of large scale multi-label categorization.The sparse representation approach we have followed in our BioASQ challenge participation [1] is essentially a large multi-label $k$-NN classifier backed by an Apache Lucene [2] index.

Our annotation scheme starts by indexing the contents of the MESINESP training articles. For each new article to be annotated, the created index index is queried using its contents as query terms. The list of similar articles returned by the indexing engine and their corresponding similarity measures are exploited to determine the following results:

[1] Source code available at https://github.com/..../mesinesp2.
[2] https://lucene.apache.org/

- predicted number of descriptors to be assigned
- ranked list of predicted DeCS codes

The first aspect is a regression problem, which aims to predict the number of descriptors to be included in the final list, depending on the number of descriptors assigned to the most similar articles identified by the indexing engine and on their respective similarity scores. The other task is a multi-label classification problem, which aims to predict a descriptors list based on the descriptors manually assigned to the most similar MESINESP articles. In both cases, regression and multi-label classification, similarity scores calculated by the indexing engine are exploited. Query terms employed to retrieve the similar articles are extracted from the original article contents and linked using a global OR operator to conform the final query sent to the indexing engine.

In our case, the scores provided by the indexing engine are actually similarity measures computed according to the weighting scheme being employed, which do not have an uniform and predictable upper bound and do not behave like a real distance. In order to ensure these similarity scores own the properties of a real distance metric, we have applied a normalization procedure, where the most similar document retrieved from the index will have a new score close to 0.0 and the scores of the rest of similar documents are adjusted in accordance with it.

With this information the number of descriptors to be assigned to the article being annotated is predicted using a weighted average scheme, where the weight of each similar article is the inverse of normalized distance cubed, that is, $\frac{1}{d^3}$.

To create the ranked list of descriptors a distance weighted voting scheme is employed, associating the same weight values (the inverse of normalized distances cubed) to the respective similar articles. Since this is actually a multi-label categorization task, there are as many voting tasks as candidate descriptors were extracted from the articles retrieved by the indexing engine. For each candidate label, positive votes come from similar articles annotated with it and negative votes come from articles not including it.

## 2.1. Text representations

Regarding article representation we have evaluated several index term extraction approaches. Our aim was to determine whether linguistic motivated index term extraction could help to improve annotation performance in the $k$-NN based method we have described. We employed the following methods:

**Stemming based representation (STEMS).** This was the simplest approach which employs stop-word removal, using a standard stop-word list for Spanish, and the default Spanish stemmer from the Snowball project[3].

**Morphosyntactic based representation (LEMMAS).** In order to deal with morphosyntactic variation in Spanish we have employed a lemmatizer to identify lexical roots and we also replaced stop-word removal with a content-word selection procedure based on part-of-speech (PoS) tags.

[3] http://snowball.tartarus.org

We have delegated the linguistic processing tasks to the tools provided by the spaCy Natural Language Processing (NLP) toolkit [4]. In our case we have employed the PoS tagging and lemmatization information provided by spaCy, using the standard Spanish models without any specific data for biomedical related contents.

Only lemmas from tokens tagged as a noun, verb, adjective, adverb or as unknown words are taken into account to constitute the final article representation, since these PoS are considered to carry the sentence meaning.

**Nominal phrases based representation (NPS).** In order to evaluate the contribution of more powerful NLP techniques, we have employed a surface parsing approach to identify syntactic motivated nominal phrases from which meaningful multi-word index terms could be extracted.

Noun Phrase (NP) chunks identified by spaCy are selected and the lemmas of the constituent tokens are joined together to create a multi-word index term.

**Dependencies based representation (DEPS).** We have also employed as index terms triples of dependence-head-modifier extracted by the dependency parser provided by spaCy. In our case spaCy provides a dependency parsing model for Spanish that identify syntactic dependency labels following the Universal Dependencies(UD) scheme. The complex index terms were extracted from the following UD relationships [5]: *acl*, *advcl*, *advmod*, *amod*, *ccomp*, *compound*, *conj*, *csuj*, *dep*, *flat*, *iobj*, *nmod* , *nsubj*, *obj*, *xcomp*, *dobj* and *pobj*.

**Named entities representation (NERS).** Another type of multi-word representations taken into account are named entities. We have employed the NER module in spaCy to extract general named entities (LOCATION,MISC , ORGANIZATION, PERSON) from articles content. We also added to this representation the set of named entities (DISEASE, MEDICATION, PROCEDURE, SYMPTOM) made available as additional resources by the MESINESP organizers.

**Keywords representation (KEYWORDS).** The last kind of multi-word representation we have included are keywords extracted with statistical methods from articles textual content. We have employed the implementation of TextRank algorithm [4] provided by the textacy library [6].

**Exact matches of DeCS labels (MATCHES).** In addition to these representations, we also have employed a pattern matching approach to extract exact matches of DeCS labels and of their corresponding synonyms from the abstract text. In our case we have added to the document representation as index term each one of those matches in order to maintain its absolute occurrence frequency.

[4] Available at https://spacy.io/

[5] Detailed list of UD relationships available at https://universaldependencies.org/u/dep/

[6] https://textacy.readthedocs.io

**Table 1**
Performance comparison of term extraction approaches in sparse representations.

| | $k$ | MiF | MiP | MiR | MaF | MaP | MaR | Acc |
|---|---|---|---|---|---|---|---|---|
| ALL | 5 | 0.3830 | 0.4030 | 0.3650 | 0.2625 | 0.3718 | 0.2640 | 0.2502 |
| | 10 | 0.4011 | 0.4230 | 0.3814 | **0.2727** | 0.4275 | **0.2724** | 0.2633 |
| | 20 | **0.4117** | **0.4349** | **0.3909** | 0.2592 | 0.5206 | 0.2571 | 0.2684 |
| | 30 | 0.4110 | 0.4343 | 0.3901 | 0.2690 | 0.4902 | 0.2667 | **0.2690** |
| | 40 | 0.4052 | 0.4283 | 0.3845 | 0.2424 | **0.5253** | 0.2416 | 0.2617 |
| STEMS | 5 | 0.3768 | 0.3976 | 0.3581 | 0.2579 | 0.3686 | 0.2577 | 0.2464 |
| | 10 | 0.3985 | 0.4210 | 0.3784 | **0.2683** | 0.4246 | **0.2667** | 0.2618 |
| | 20 | **0.4065** | **0.4302** | **0.3854** | 0.2619 | 0.4759 | 0.2588 | **0.2654** |
| | 30 | 0.4059 | 0.4297 | 0.3846 | 0.2467 | 0.5032 | 0.2445 | 0.2635 |
| | 40 | 0.4032 | 0.4264 | 0.3823 | 0.2391 | **0.5217** | 0.2380 | 0.2601 |
| LEMMAS | 5 | 0.3762 | 0.3963 | 0.3581 | 0.2526 | 0.3656 | 0.2538 | 0.2454 |
| | 10 | 0.3963 | 0.4181 | 0.3766 | **0.2656** | 0.4222 | **0.2658** | 0.2595 |
| | 20 | **0.4057** | **0.4280** | **0.3855** | 0.2621 | 0.4765 | 0.2599 | **0.2648** |
| | 30 | 0.4045 | 0.4271 | 0.3841 | 0.2483 | 0.5067 | 0.2473 | 0.2616 |
| | 40 | 0.4009 | 0.4235 | 0.3807 | 0.2399 | **0.5274** | 0.2388 | 0.2580 |
| NERS | 5 | 0.2811 | 0.3080 | 0.2584 | **0.1682** | 0.2675 | **0.1690** | 0.1712 |
| | 10 | 0.2974 | 0.3270 | 0.2727 | 0.1681 | 0.3149 | 0.1670 | 0.1805 |
| | 20 | **0.3072** | **0.3386** | **0.2811** | 0.1612 | 0.3687 | 0.1602 | **0.1847** |
| | 30 | 0.3064 | 0.3381 | 0.2801 | 0.1542 | 0.3950 | 0.1526 | 0.1826 |
| | 40 | 0.3033 | 0.3348 | 0.2772 | 0.1487 | **0.4091** | 0.1484 | 0.1791 |
| KEYWORDS | 5 | 0.3346 | 0.3514 | 0.3194 | 0.1991 | 0.3027 | **0.2001** | 0.2164 |
| | 10 | 0.3507 | 0.3689 | 0.3342 | **0.2010** | 0.3460 | 0.1994 | 0.2261 |
| | 20 | 0.3580 | 0.3770 | 0.3408 | 0.1952 | 0.3992 | 0.1908 | **0.2300** |
| | 30 | **0.3592** | **0.3789** | **0.3415** | 0.1863 | 0.4454 | 0.1805 | 0.2290 |
| | 40 | 0.3579 | 0.3775 | 0.3402 | 0.1749 | **0.4743** | 0.1684 | 0.2265 |
| NPS | 5 | 0.2111 | 0.2497 | 0.1828 | **0.0895** | 0.1735 | **0.0875** | 0.1185 |
| | 10 | 0.2257 | 0.2681 | 0.1949 | 0.0889 | 0.2145 | 0.0853 | 0.1265 |
| | 20 | **0.2305** | **0.2744** | **0.1987** | 0.0811 | 0.2571 | 0.0756 | **0.1273** |
| | 30 | 0.2289 | 0.2728 | 0.1971 | 0.0713 | 0.2697 | 0.0659 | 0.1247 |
| | 40 | 0.2282 | 0.2723 | 0.1964 | 0.0650 | **0.2811** | 0.0596 | 0.1232 |
| DEPS | 5 | 0.3483 | 0.3648 | 0.3332 | 0.2138 | 0.3144 | **0.2170** | 0.2261 |
| | 10 | 0.3630 | 0.3808 | 0.3468 | **0.2180** | 0.3633 | **0.2170** | 0.2359 |
| | 20 | **0.3702** | **0.3899** | **0.3524** | 0.2122 | 0.4167 | 0.2095 | **0.2385** |
| | 30 | 0.3654 | 0.3849 | 0.3479 | 0.1981 | 0.4444 | 0.1939 | 0.2325 |
| | 40 | 0.3628 | 0.3823 | 0.3452 | 0.1894 | **0.4681** | 0.1840 | 0.2292 |
| MATCHES | - | 0.2574 | 0.2016 | 0.3559 | 0.3171 | 0.3815 | 0.3674 | 0.1517 |

## 2.2. Inverted DeCS code profiles

Apache Lucene provides a general information retrieval engine that implements a vector space model with different well-known scoring algorithms, such as TF-IDF, BM25 variants, and others. Lucene maintains an inverted index where it links the index terms extracted by its analyzers with the documents where they appear and maintains information about occurrence frequencies of these index terms in order to calculate the query scores.

As a complementary experiment to our proposal of sparse similarity, instead of using a conventional retrieval system we have proposed our own simplified version of an inverted

index at descriptor level. Each possible index term is linked to a list of DeCS codes with which it maintains a degree of co-correlation greater than a certain threshold. The intuition behind this approach is that the presence of certain indexing terms in a given document is a good predictor of the convenience of labeling that document with DeCS codes strongly linked, from a co-occurrence point of view, with those terms.

To implement this idea we have used as a co-occurrence metric between index terms and DeCS codes the Normalized Pointwise Mutual Information (NPMI), calculated on the training set as follows, being $t$ an index term and $d$ a DeCS code:

$$NPMI(t,c) = \frac{PMI(t,d)}{-log(P(t,c))}$$

where $PMI$ is the Pointwise Mutual Information computed by

$$PMI(t,c) = log\left(\frac{P(t,c)}{P(t)\cdot P(c)}\right)$$

with $P(t,c) = \frac{|\text{docs. labeled with } c \text{ containing term } t|}{|\text{docs. in training collection}|}$, $P(t) = \frac{|\text{docs. containing term } t|}{|\text{docs. in training collection}|}$ and $P(c) = \frac{|\text{docs. labeled with } c|}{|\text{docs. in training collection}|}$.

The measure $NPMI(t,c)$ normalizes the values of $PMI(t,c)$ in $[-1,1]$, resulting in $-1$ for a term $t$ and a DeCS code $c$ never occurring together, 0 for independence, and $+1$ for complete co-occurrence of term $t$ and code $c$.

For the construction of these inverse DeCS code profiles, we have treated separately the single index terms, corresponding to representations of type LEMMAS, and the compound index terms, which correspond to the multi-word terms extracted by NERS, NPS and KEYWORDS representations. As thresholds for the NPMI co-occurrence metric we have used the values 0.25, 0.50 and 0.75, linking with each index term, both single and compound, the DeCS codes whose co-occurrence measured according to NPMI exceeds these thresholds.

With these inverted descriptor profiles we have implemented a simple matching scheme to annotate an input document. Given a document to be annotated, its simple and compound terms are extracted, using the methods described in the preceding section. Using the described term-to-code profiles, the NPMI co-occurrence scores of each possible candidate DeCS code are accumulated in a table every time one of the terms related to a given DeCS code appears.

To build the final list of DeCS code candidates to be assigned to a given test document we use as a reference the set of codes predicted by the sparse similarity method described in the previous section. This reference set determines the number of DeCS codes to predict, $n$, and provides additional codes needed to fulfill that number of output codes whether the number of DeCS codes with higher accumulated co-occurrence scores predicted with the DeCS codes profiles are less than $n$.

## 3. Similarity with dense representations

In recent years we are experiencing the rise of powerful language models such as BERT and other similar approaches that have increased the performance of multiple language processing tasks and have allowed that solutions based on Transformer models to dominate the state-of-the-art in many NLP fields today. A natural evolution of these word embeddings is to

**Table 2**
Performance results with inverse DeCS code profiles.

| terms type | threshold | MiF | MiP | MiR | MaF | MaP | MaR | Acc |
|---|---|---|---|---|---|---|---|---|
| single | 025 | 0.1707 | 0.1803 | 0.1620 | 0.1593 | 0.2692 | 0.1752 | 0.0973 |
| | 050 | 0.2860 | 0.3022 | 0.2716 | 0.2192 | 0.2906 | 0.2594 | 0.1716 |
| | 075 | 0.4147 | 0.4381 | 0.3937 | 0.2858 | 0.4694 | 0.2990 | 0.2689 |
| compound | 025 | 0.2174 | 0.2297 | 0.2064 | 0.1958 | 0.2413 | 0.2285 | 0.1276 |
| | 050 | 0.3052 | 0.3224 | 0.2897 | 0.2557 | 0.2713 | 0.2939 | 0.1870 |
| | 075 | **0.4247** | **0.4486** | **0.4032** | 0.2838 | **0.4959** | 0.2901 | **0.2768** |
| both | 025 | 0.2417 | 0.2553 | 0.2295 | 0.2224 | 0.3025 | 0.2492 | 0.1432 |
| | 050 | 0.3183 | 0.3362 | 0.3021 | 0.2694 | 0.3157 | **0.3126** | 0.1960 |
| | 075 | 0.4191 | 0.4427 | 0.3979 | **0.2950** | 0.4554 | 0.3123 | 0.2721 |

move them towards embeddings at the sentence-level with approaches as those provided by SentenceTransformers [6] project [7] that allows to convert sentences in natural languages into dense vectors with enriched semantics.

In this context we have evaluated the possibility of taking advantage of these dense semantic representations of whole sentences as a basis for an approach similar to the one described in the previous section. We replace the use of text indexers to match similar documents with the search for similar vectors in the dense vector space where documents from the training dataset are represented. The procedure that we follow to generate the dense vectors that will represent a document as a whole, either from training or test collections, is the following:

- The paragraphs of the document are split into sentences and the dense vector that represents every sentence is calculated using Sentence Transformers models.
- The dense representation of the whole document is calculated as the mean vector of the dense vectors extracted from the sentences that the abstract is comprised of.

Once we have the dense representations of the training documents using this procedure, we use the FAISS [3] library [8] to create a searchable index on these dense vectors. This index allows us to efficiently calculate distances between dense vectors and determine for the dense vector associated with a given test abstract (our query vector) the list of $k$ closest training dense vectors using the Euclidean distance or other similarity metrics on vectors.

By having this mechanism of similarity between dense vectors, the procedure used to annotate the test documents is analogous to that one used with the sparse similarity approach with Lucene indices. In this case we can directly use the real distances between the query vector generated from the text to be annotated and the most similar $k$ dense vectors provided by FAISS library. With these distances, the number of labels to be assigned is estimated and the output DeCS codes are selected by means of the weighted voting scheme already described in section 2.

[7] https://www.sbert.net/
[8] https://github.com/facebookresearch/faiss

**Table 3**
Performance results with dense representations.

| | $k$ | MiF | MiP | MiR | MaF | MaP | MaR | Acc |
|---|---|---|---|---|---|---|---|---|
| MONO | 5 | 0.2818 | 0.2916 | 0.2726 | 0.1390 | 0.2234 | 0.1415 | 0.1790 |
| | 10 | 0.3019 | 0.3130 | 0.2915 | 0.1470 | 0.3364 | 0.1446 | 0.1917 |
| | 20 | 0.3097 | 0.3206 | 0.2995 | 0.1477 | 0.4344 | 0.1419 | 0.1965 |
| | 30 | 0.3103 | 0.3213 | 0.3000 | 0.1466 | 0.4872 | 0.1394 | 0.1969 |
| | 40 | 0.3103 | 0.3213 | 0.3000 | 0.1461 | 0.5208 | 0.1377 | 0.1971 |
| MULTI | 5 | 0.3443 | 0.3595 | 0.3302 | 0.1861 | 0.2926 | 0.1883 | 0.2221 |
| | 10 | 0.3642 | 0.3793 | 0.3504 | **0.1970** | 0.3744 | 0.1961 | 0.2373 |
| | 20 | 0.3716 | 0.3869 | 0.3574 | 0.1948 | 0.4377 | **0.1931** | 0.2420 |
| | 30 | **0.3731** | **0.3879** | **0.3593** | 0.1935 | 0.4780 | 0.1904 | **0.2429** |
| | 40 | 0.3709 | 0.3853 | 0.3574 | 0.1880 | **0.4937** | 0.1841 | 0.2408 |

## 4. Premilinary results

In this section we briefly present the results of a series of preliminary experiments carried out to validate the methods described in the previous sections and to characterize the parameters to be used in our official runs submitted to the challenge. All of these experiments have used the data provided by the organization of the MESINESP2 challenge for sub-track-1 [2], with a train dataset with 237,574 articles annotated with DeCS codes and a development dataset with 1,065 documents.

In the case of assigning DeCS codes using similarity over sparse representations supported by an Apache Lucene index, we have separately evaluated the performance of the different methods of extraction of index terms introduced in section 2. We also tested different values for the parameter $k$, the number of neighbors considered to predict the number of labels and to vote the final list of output labels. Table 1 shows the results obtained in this previous evaluation.

Regarding the use of the inverse profiles of DeCS codes, we have evaluated the use of simple index terms, compound index terms and a mixture of both to build the DeCS codes profiles. Using in all cases the three co-occurrence thresholds previously indicated, 0.25, 0.5, 0.75. To determine the number of DeCS codes to predict in each test document and to provide additional codes, the result list from best execution of the sparse similarity scheme in Table 1 has been used as reference. The results of these experiments with inverse profiles are shown in Table 2.

Finally, in the case of assigning DeCS codes through similarity over dense representations, we have evaluated the use of Sentence Transformers with two different pretrained language models, one multilingual model [9] and a Spanish monolingual model [10]. We also evaluated different values for the $k$ parameter. The obtanied results are detailed in Table 3.

[9]Using pretrained sentence-level model `stsb-xlm-r-multilingual` from Sentence Transformers project. Provides dense vectors with 768 dimensions.

[10]Using pretrained word-level model `mrm8488/electricidad-base-generator` from Hugging Face models repository. Provides dense vectors with 256 dimensions.

**Table 4**
Official results for BioASQ MESINESP2 Task.

**Sub-track 1**

| system | rank | MiF | EBP | EBR | EBF | MaP | MaR | MaF | MiP | MiR | Acc. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| best | 1/26 | 0.4837 | 0.5077 | 0.4736 | 0.4763 | 0.5237 | 0.3990 | 0.3926 | 0.5077 | 0.4618 | 0.3261 |
| iria-mix | 15/26 | 0.3725 | 0.4245 | 0.3402 | 0.3662 | 0.5345 | 0.2326 | 0.2354 | 0.4193 | 0.3351 | 0.2341 |
| iria-4 | 17/26 | 0.3656 | 0.3938 | 0.3481 | 0.3585 | 0.4476 | 0.2877 | 0.2760 | 0.3909 | 0.3435 | 0.2279 |
| iria-1 | 18/26 | 0.3406 | 0.3670 | 0.3238 | 0.3339 | 0.4236 | 0.2348 | 0.2315 | 0.3641 | 0.3199 | 0.2089 |
| iria-2 | 19/26 | 0.3389 | 0.3650 | 0.3218 | 0.3319 | 0.4214 | 0.2327 | 0.2293 | 0.3622 | 0.3185 | 0.2073 |
| MESINESP baseline | 20/26 | 0.2876 | 0.2449 | 0.3839 | 0.2841 | 0.3720 | 0.3787 | 0.3438 | 0.2335 | 0.3746 | 0.1710 |
| iria-3 | 21/26 | 0.2537 | 0.2758 | 0.2337 | 0.2460 | 0.2869 | 0.0854 | 0.0817 | 0.2729 | 0.2369 | 0.1480 |

**Sub-track 2**

| system | rank | MiF | EBP | EBR | EBF | MaP | MaR | MaF | MiP | MiR | Acc. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| best | 1/21 | 0.3640 | 0.3666 | 0.3655 | 0.3558 | 0.4177 | 0.3391 | 0.3102 | 0.3666 | 0.3614 | 0.2242 |
| iria-1 | 12/21 | 0.2454 | 0.2303 | 0.2625 | 0.2379 | 0.3167 | 0.1863 | 0.1534 | 0.2289 | 0.2644 | 0.1411 |
| iria-4 | 14/21 | 0.2003 | 0.1919 | 0.2142 | 0.1958 | 0.1620 | 0.2049 | 0.1571 | 0.1868 | 0.2158 | 0.1132 |
| iria-mix | 15/21 | 0.2003 | 0.1919 | 0.2142 | 0.1958 | 0.1620 | 0.2049 | 0.1571 | 0.1868 | 0.2158 | 0.1132 |
| iria-3 | 16/21 | 0.1562 | 0.1422 | 0.1681 | 0.1502 | 0.1617 | 0.0730 | 0.0505 | 0.1419 | 0.1736 | 0.0857 |
| MESINESP baseline | 17/21 | 0.1288 | 0.0971 | 0.3791 | 0.1452 | 0.0977 | 0.3619 | 0.2403 | 0.0781 | 0.3678 | 0.0802 |

**Sub-track 3**

| system | rank | MiF | EBP | EBR | EBF | MaP | MaR | MaF | MiP | MiR | Acc. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| best | 1/21 | 0.4514 | 0.4487 | 0.4662 | 0.4494 | 0.5041 | 0.4271 | 0.4138 | 0.4487 | 0.4541 | 0.3005 |
| iria-2 | 7/21 | 0.3203 | 0.3509 | 0.2878 | 0.3061 | 0.4980 | 0.3166 | 0.3171 | 0.3657 | 0.2849 | 0.1910 |
| MESINESP baseline | 8/21 | 0.2992 | 0.4117 | 0.2298 | 0.2827 | 0.5290 | 0.2497 | 0.2518 | 0.4293 | 0.2296 | 0.1779 |
| iria-mix | 13/21 | 0.2542 | 0.2790 | 0.2414 | 0.2528 | 0.4659 | 0.2284 | 0.2213 | 0.2750 | 0.2364 | 0.1526 |
| iria-4 | 16/21 | 0.2169 | 0.2251 | 0.2085 | 0.2119 | 0.3105 | 0.2442 | 0.2289 | 0.2232 | 0.2109 | 0.1288 |
| iria-1 | 18/21 | 0.1871 | 0.1941 | 0.1826 | 0.1844 | 0.2589 | 0.1966 | 0.1825 | 0.1926 | 0.1820 | 0.1093 |
| iria-3 | 19/21 | 0.0793 | 0.0824 | 0.0758 | 0.0777 | 0.1120 | 0.0598 | 0.0501 | 0.0822 | 0.0765 | 0.0437 |

## 5. Official runs and discussion

Although our team has submitted results to all the sub-tracks of the MESINESP challenge, a parameterization adapted to the specific characteristics of each sub-track has not been carried out. All the configurations used in the official runs have been identical in the three sub-tracks only with adjustments in the number of neighbors considered according to the results of previous experiments with the provided development datasets. The only exception is sub-track-3 where a substantially different configuration has been used in one of the submitted runs.

In table 4 the official performance measures obtained by our runs in the three MESINESP2 sub-tasks are shown. The official runs submitted during our participation were created using the following configurations:

**iria1.** This run followed the sparse similarity approach described in section 2. The sparse representation of MESINESP articles was created using all of the index term extraction methods described in section 2.1. During indexing and querying, terms appearing in 5 or

less abstracts and terms used in more than 50% of training documents were discarded. The number of neighbors used by the $k$-NN classifier was 20 and the predicted number of descriptors to be returned was increased by a 10% in order to ensure slightly better values in recall related measures.

**iria2.** For this run in Sub-track-1 the same setup as `iria1` was employed, but instead of using the original train dataset this run applied a sort of Label Powerset approach proposed in our previous participation at MESINESP challenge [8]. A new training dataset annotated with those "metalabels" was created by joining pairs of DeCS labels with NPMI co-occurrence scores above 0.25. This dataset was indexed and processed as described for run `iria1`.

In Sub-track-3 `iria2` setup followed the inverse DeCS codes profile approach from section 2.2. The employed profiles were a mix of single and compound profiles with a threshold of 0.75 for the co-occurrence scores. Instead of using the results of a sparse method as reference this runs was created directly over the set of exact matches extracted from the abstract text (MATCHES representation).

**iria3.** This run followed the dense similarity approach introduced in section 3. We employed the multilingual word model to create dense vectors for every training document and indexed those vectors in a FAISS index. The number of neighbors used by the $k$-NN classifier was 30 and, as in `iria1` run, the predicted number of descriptors to be returned was increased by a 10%.

**iria4.** This run employed the inverse DeCS codes profile approach introduced in section 2.2. The employed profiles were a mix of single and compound profiles created using a threshold of 0.75 for the co-occurrence scores between terms and DeCS codes. The reference results employed by this approach were those from `iria1`.

**iria-mix.** This run mixed the predictions of `iria1` and the predictions of `iria3`, adding the exact matches extracted from the textual content of the labeled abstract (MATCHES representation). Predictions from `iria1` and `iria3` had a weight of 1.0 and the DeCS labels matched on the abstract text were weighted by 1.5.

The results of our participation in the MESINESP task of the BioASQ biomedical semantic indexing challenge were not very competitive, far from the performance of the winner teams. In any case, we think that our experience confirms the suitability of methods based on similarity as a viable alternative for large scale text categorization in rich domains such as biomedical document collections.

We have evaluated different classification methods based on similarity over sparse and dense representations. In our experiments the best results have been obtained using sparse representations where different index term extraction techniques were combined. That confirmed the results in our previous BioASQ participation, with small performance improvements mainly due to improvements in the quality of the employed NLP tools and models.

Results with similarity over dense representations were generally disappointing. The proposed method, a simple mean of sentence based dense vectors, was extremely simple and it

remains for future to evaluate better approaches that could improve the dense representation of the documents as a whole. In the same way, it is expected that fine-tuning the language models using biomedical texts will improve their performance and this is precisely one of the lines of future work to experiment with.

## Acknowledgments

F.J. Ribadas-Pena and S. Cao have been supported by ERDF/MICINN-AEI (TIN2017-85160-C2-2-R and PID2020-113230RB-C22), and by the Galician Regional Government (Xunta de Galicia) under projects ED431D-2018/50 and ED431D-2017/12.
E. Kuriyozov received funding from ERDF/MICINN-AEI (ANSWER-ASAP, TIN2017-85160-C2-1-R, and SCANNER-UDC, PID2020-113230RB-C21), from Xunta de Galicia (ED431C 2020/11) and from Centro de Investigación de Galicia "CITIC", funded by Xunta de Galicia and the European Union (European Regional Development Fund- Galicia 2014-2020 Program), by grant ED431G 2019/01. He is also funded for his PhD by El-Yurt-Umidi Foundation under the Cabinet of Ministers of the Republic of Uzbekistan.

## 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] 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). Zenodo, URL: https://doi.org/10.5281/zenodo.4722925.

[3] J. Johnson, M. Douze, H. Jégou: Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734, 2017, URL: https://arxiv.org/abs/1702.08734.

[4] R. Mihalcea, P. Tarau: TextRank: Bringing order into texts. Association for Computational Linguistics. 2004.

[5] 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

[6] N. Reimers, I. Gurevych: Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2020.

[7] F.J. Ribadas-Pena, L.M. de Campos, V.M. Darriba-Bilbao, A. E. Romero: CoLe and UTAI at BioASQ 2015: Experiments with Similarity Based Descriptor Assignment. CEUR Workshop Proceedings, vol. 1391. 2015.

[8] F.J. Ribadas-Pena, S. Cao, E. Kuriyozov: CoLe and LYS at BioASQ MESINESP8 Task: Similarity based Descriptor Assigment in Spanish. CEUR Workshop Proceedings, vol. 2696. 2020.