Universidad Politécnica
de Madrid

Escuela Técnica Superior de
Ingenieros Informáticos

Máster Universitario en Inteligencia Artificial

Trabajo Fin de Máster

Named Entity Recognition of Human Phenotype Ontology Concepts in Spanish Clinical Texts

Autor: Luis Couto Seller

Tutores: Miguel García Remesal y Raúl Alonso Calvo

Madrid, Julio 2024

Este Trabajo Fin de Máster se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Máster
Máster Universitario en Inteligencia Artificial
Título: Named Entity Recognition of Human Phenotype Ontology Concepts in Spanish Clinical Texts
Julio 2024

Autor: Luis Couto Seller

Tutores:
Miguel García Remesal
Departamento de Inteligencia Artificial
ETSI Informáticos
Universidad Politécnica de Madrid

Raúl Alonso Calvo
Departamento de Lenguajes, Sistemas Informáticos e Ingeniería del Software
ETSI Informáticos
Universidad Politécnica de Madrid

Resumen

El presente trabajo aborda el reconocimiento de entidades de la Human Phenotype Ontology (HPO) en textos clínicos en español, utilizando técnicas avanzadas de Procesamiento del Lenguaje Natural (NLP). Este proceso es fundamental para la estandarización y codificación de información fenotípica contenida en la Historia Clínica Electrónica (HCE), facilitando la interoperabilidad semántica entre sistemas de salud y mejorando la precisión en el diagnóstico de enfermedades complejas.

Con la digitalización de los datos clínicos, ha surgido la necesidad de utilizar terminologías y ontologías clínicas para asegurar la interoperabilidad de la información, contexto en el cual la HPO es crucial, ya que permite identificar patrones y correlaciones entre síntomas y enfermedades, agilizando y mejorando la precisión diagnóstica, especialmente en enfermedades raras y complejas.

El principal desafío radica en que la mayoría de la información fenotípica se registra en formato de texto libre, lo que dificulta su estandarización automática. Además, existe una escasez de datos etiquetados en español, ya que la HPO no está completamente traducida a este idioma, lo que limita el uso de modelos de aprendizaje profundo. Este proyecto se propone desarrollar un modelo híbrido que combine técnicas de búsqueda en diccionarios y modelos de aprendizaje profundo para mejorar el reconocimiento de términos HPO en textos clínicos en español.

La arquitectura del modelo se basa en dos módulos principales: el de búsqueda en diccionarios y el modelo de aprendizaje profundo. Ambos utilizan un diccionario basado en la HPO como fuente de conocimiento. Los pasos clave llevados a cabo incluyen:

1. Construcción del diccionario.
2. Entrenamiento del modelo de aprendizaje profundo.
3. Preprocesamiento de los textos de entrada.
4. Etiquetado dual del texto mediante métodos basados en aprendizaje profundo y diccionarios.
5. Combinación de resultados

Adicionalmente, para aumentar el número de instancias en el diccionario, se emplearon técnicas de aumento de datos, como la traducción inversa (round-trip translation).

El sistema híbrido demostró una mejora significativa en el recall, reconociendo variaciones no incluidas en el diccionario y manteniendo una precisión similar a la de la búsqueda en diccionarios. Los resultados finales del modelo muestran una precisión de 0.7016, un recall de 0.7655 y un F1 score de 0.7321, lo que refleja un equilibrio entre precisión y capacidad de reconocimiento de conceptos.

Este trabajo sienta las bases para la implementación de sistemas automatizados de reconocimiento de entidades fenotípicas en español, contribuyendo a la mejora de la interoperabilidad semántica en el ámbito de la salud y potenciando la capacidad diagnóstica en entornos clínicos.

Abstract

The present work addresses the recognition of entities from the Human Phenotype Ontology (HPO) in Spanish clinical texts, using advanced Natural Language Processing (NLP) techniques. This process is fundamental for the standardization and coding of phenotypic information in electronic health records (EHR), facilitating semantic interoperability between health systems and improving the precision in diagnosing complex diseases.

With the digitization of clinical data, the need to use clinical terminologies and ontologies has arisen to ensure information interoperability. The HPO is crucial in this context, as it allows the identification of patterns and correlations between symptoms and diseases, expediting and improving diagnostic precision, especially in rare and complex diseases.

The main challenge lies in the fact that most phenotypic information is recorded in free-text format, making automatic standardization difficult. Additionally, there is a scarcity of labeled data in Spanish since the HPO is not fully translated into this language, limiting the use of deep learning models. This project aims to develop a hybrid model that combines dictionary-based search techniques and deep learning models to improve the recognition of HPO terms in Spanish clinical texts.

The model architecture is based on two main modules: the dictionary search module and the deep learning model. Both use a dictionary based on the HPO as a knowledge source. The key steps include:

1. Dictionary construction.
2. Training the deep learning model.
3. Preprocessing input texts.
4. Dual text labeling using deep learning and dictionary-based methods.
5. Combining results.

Additionally, to increase the number of instances in the dictionary, data augmentation techniques such as round-trip translation were employed.

The hybrid system demonstrated a significant improvement in recall, recognizing variations not included in the dictionary while maintaining precision similar to dictionary search. The final model results show a precision of 0.7016, a recall of 0.7655, and an F1 score of 0.7321, reflecting a balance between precision and concept recognition ability.

This work lays the foundation for the implementation of automated phenotypic entity recognition systems in Spanish, contributing to the improvement of semantic interoperability in the healthcare field and enhancing diagnostic capabilities in clinical settings.

Table of contents

Index of Figures

Index of Tables

1 Introduction

In recent years, clinical data have evolved thanks to its digitization, which has laid the foundations for the emergence of the Electronic Health Record (EHR), allowing the exploitation of data for secondary purposes such as the exchange of information between healthcare institutions or the planning of clinical trials, with interoperability being a prerequisite for this. It is in this context that the need arises to use clinical terminologies and ontologies, which play a vital role in the semantic interoperability of information.

Data standardization with clinical ontologies, such as the Human Phenotype Ontology (HPO), offers significant value in phenotypic data management and analysis. These ontologies allow accurate coding of phenotypes, which facilitates the identification of patterns and correlation of symptoms with possible diseases and their associated genes, especially in the case of rare and complex diseases this improves diagnostic precision, and also optimizes treatment planning and patient follow-up. In addition, the ontology promotes interoperability between different healthcare systems, enabling more effective information exchange.

The exploitation of medical data from the EHRs thanks to the data standardization processes from different institutions and hospitals has been one of the main motivations for the development of computer tools that facilitate this type of standardization process. These tools are designed to Extract, Transform and Load (ETL) the clinical data into standardized formats, ensuring that the data can be used across different platforms and research studies. This enhances the ability of clinicians and researchers to access high-quality interoperable data that can drive advancements in medical science and patient care.

The HPO is particularly significant in this context due to its clear structure and detailed annotations. HPO has over 18,000 terms that describe human phenotypic abnormalities and their relationships, where each term in HPO is linked to related genes, diseases, and other relevant clinical information, creating a robust framework for clinical and research applications. This extensive hierarchy of terms allows for precise phenotype-genotype correlations, helping in the identification of genetic diseases.

The use of HPO can lead to more personalized patient care, documenting patient phenotypes, healthcare providers can better understand the patient's condition, leading to more precise treatment plans. For example, in the case of genetic disorders, HPO can help clinicians identify the exact nature of the phenotypic manifestations, which is crucial for making accurate diagnoses and predicting disease progression. Furthermore, HPO facilitates the matching of patients to

clinical trials, ensuring that individuals who are most likely to benefit from experimental therapies are identified and enrolled.

From a research perspective, HPO allows for the aggregation and comparison of data across different studies, enhancing the power of genetic research. Researchers can use HPO to integrate data from diverse sources, such as genomic studies and clinical records, enabling large-scale analyses that can uncover new insights into disease mechanisms and potential therapeutic targets. This is particularly valuable in the study of rare diseases, where patient data is often sparse and dispersed across multiple institutions.

The integration of HPO into EHRs and clinical processes represents a significant advancement in the field of biomedical informatics, it bridges the gap between raw clinical data and meaningful insights, facilitating improved patient care, more effective clinical research, and a deeper understanding of human diseases.

This project aims to develop a tool that can perform automatic labeling of HPO phenotypes in Spanish clinical free text using a hybrid model that combines dictionary search techniques and deep learning models, trying to achieve results like those achieved by certain state-of-the-art works that are designed in the context of English.

1.1 Objectives

The main objective of this work is to develop a hybrid NLP model that combines dictionary search techniques and deep learning models to improve the recognition of HPO terms in Spanish clinical texts. The specific objectives include:

1. **Construction of a dictionary based on HPO**: Create a dictionary that includes labels, synonyms, and lemmas of HPO concepts in Spanish.

2. **Training a deep learning model**: Develop and train a model that can recognize not only the phenotypes described in the dictionary but also variants not included in it.

3. **Preprocessing clinical texts**: Prepare the input texts for the subsequent tagging.

4. **Hybrid text labeling**: Implement a dual labeling system that combines the results obtained from both methods to improve precision and recall in HPO term recognition.

5. **Data augmentation**: Apply data augmentation techniques to increase the number of synonyms per HPO concept and improve the model's performance.

1.2 Structure

To provide a complete and detailed overview of the development and evaluation process this document is structured as follows:

1. Introduction: Context, objectives and structure of the work are presented.

2. State of the Art: Review of existing techniques and methods for NER in the biomedical field, including dictionary-based methods and deep learning models.

3. Methodology: Description of the architecture of the proposed model, the processes of dictionary construction, model training, text preprocessing, and combination of results.

4. Results: Results obtained with the hybrid model and its analysis, comparing its performance with methods based only on dictionaries or deep learning.

5. Conclusions and Future Lines: Final conclusions of the work and future lines of research proposed to improve and extend the model developed.

6. Bibliography: Bibliographic references used throughout the work.

2 State of the art

2.1 Named Entity Recognition

Biomedical Named Entity Recognition (BioNER) is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured medical text into predefined categories such as proteins, genes, phenotype, medical procedures, drugs, diseases, etc. Essentially, NER aims to identify and classify key information elements within text into distinct categories that are relevant for processing natural language. This process is crucial for a wide range of natural language processing (NLP) applications, such as information retrieval, question answering, content classification, knowledge extraction, and relationship extraction, among others. It helps in structuring and categorizing data within texts, making it easier for algorithms and systems to understand, process, and respond to human language in a meaningful way.

In the biomedical context, the automatic identification of phenotypes in medical texts remains a challenge. Numerous supervised and distantly supervised methods are proposed in the literature to address BioNER tasks. Existing approaches can be generally classified into three categories, rule-based (string matching, dictionary-based, statistical model, etc.) algorithms, machine learning algorithms, including recently developed deep learning methods, and hybrid models combining both approaches.

The first to appear were rule-based models, mainly those that were dictionary-based. Within this group of models are MetaMap [1], NCBO annotator [2], ClinPhen [3] and Doc2HPO [4]. MetaMap, developed by the National Library of Medicine, links biomedical texts to the Metathesaurus using a knowledge-based approach. The NCBO annotator generates direct annotations of raw text by syntactic concept recognition using a dictionary of UMLS terms and NCBO BioPortal. It then expands these annotations with knowledge of one or more ontologies. ClinPhen applies sequential analysis and a rule-based NLP system to filter true mentions from false positives. CLAMP is a clinical NLP tool that offers state-of-the-art components and a graphical interface to build customized NLP workflows, including phenotype identification.

The trend in recent years has been the application of machine learning techniques, including deep learning. The main advantages of these methods are high accuracy and no need for manual features engineering. In the literature there are several architectures and proposals applied to the BioNER problem, including convolutional neural networks (CNN), graph convolutional networks (GCN) , recurrent neural networks (RNN) and transformers. These models generally require large amounts of labeled training data, which is a major problem especially in the clinical setting. This is why the branch of research

that has focused on using hybrid deep learning models that use dictionary knowledge has emerged.

The focus of this thesis will be on models based on deep learning models combined with the use of dictionaries, in the following sections a detailed review of the different techniques and architectures used for BioNER problems will be made.

2.2 Dictionary methods

Before the widespread use of deep learning models, simpler techniques were used for NER. These techniques were based on the use of a knowledge source or dictionary containing all the entities to be recognized in the text. In their paper Alexandra Pomares et al. [5] review the ways of performing dictionary search focused on the context of clinical texts, considering the grammatical complexity that this type of text entails.

2.2.1 One to one recognition

The most basic and simple way to recognize dictionary terms is simply to look for exact matches of these words in the text. This means that there must be an exact one-to-one match between one of the dictionary words and a portion of the text. For example, let's say that G is our diagnostic dictionary and DocX is a document in the corpus:

$$G = \{'diabetes', 'hypertension', 'COPD', 'influenza'\} \quad (1)$$

DocX='John Doe is a 67 year-old diabetic white male with a history of COPD, and hypertension. Mr. Doe was hospitalized 20 days ago at High Plains Hospital for pneumonia resulting from influezna ...'

DocXAnnot='John Doe is a 67 year-old diabetic white male with a history of [**COPD**], and [**hypertension**]. Mr. Doe was hospitalized 20 days ago at High Plains Hospital for pneumonia resulting from influezna ...'

Figure 1. One to one recognition example [5]

2.2.2 Fuzzy recognition

In the context of NER for clinical EHR texts, it is very common to find orthographic and spelling errors. This seriously affects the recall of the dictionary-based NER due to the mismatch between dictionary words and text words. For example, in the dictionary the word “pneumonia” will not appear with the misspelled term “pneumonia”. One way to solve this is to use the Fuzzy Gazetteer approach [6] to be able to not only find exact matches but also those with misspellings.

The “edit distance” metric is a numerical representation of the number of characters changed between two words and is very useful for comparing text terms and dictionary terms. An acceptance limit must be defined as a proportion of the dictionary word length. For example, we will admit more changes in those

words that are long such as “hypertension” than in those that are shorter such as “COPD”.

Using the same example as the previous section, the Fuzzy Gazetteer match approach result in the following terms:

DocXAnnFuzz= ’John Doe is a 67 year-old diabetic white male with a history of [**COPD:COPD(0)**], and [**hypertension:hypertension(0)**]. Mr. Doe was hospitalized 20 days ago at High Plains Hospital for pneumonia resulting from [**influezna:influenza(2)**] ...’

Figure 2. Fuzzy recognition example [5]

2.2.3 Stem recognition

A final example of a dictionary matching technique is stem recognition. The search is based on the lemmas or stems of the dictionary, finding the stem of each word in the documents.

In the previous example, we first compute *G’*, the stemmed version of the dictionary *G* and then, document *DocX* is processed with the same stemmer to create *DocX’*.

$$G' = \{diabet', hypertens', copd', influenza'\} \quad (2)$$

DocX’= ’john doe is a 67 year-old diabet white male with a histori of copd and hypertens mr doe was hospit 20 day ago at high plain hospit for pneumonia result from influezna ...’

DocXAnnStem= ’john doe is a yearold [**diabet**] white male with a histori of [**copd**] and [**hypertens**] mr doe was hospit day ago at high plain hospit for pneumonia result from influezna...’

Figure 3. Stem recognition example [5]

2.3 Deep Learning methods

After a review of the models proposed in the literature to solve the BioNER problem, it has been observed that most deep learning architectures that use a dictionary as a knowledge base use the models explained in this section.

2.3.1 Long short-term memory (LSTM)

Long-short term memory (LSTM) belongs to the RNN family and processes sequential data. Essentially, LSTM receives a sequence of vector inputs *xt* and generates a sequence of hidden contexts h. Each *ht* represents the activation of the LSTM at time *t*. In BioNER task, each output *h* is ultimately transformed into a class label by the Conditional Random Field (CRF) layer [7]. LSTM can manage long input sequences using internal memory gates and forget gates. Given that the data consists of a sequence of word inputs, there is only one LSTM unit, and all gates within this unit are recurrently utilized. The output from time *t-1* combines with the input at time t and re-enters the node input gate, creating a loop.

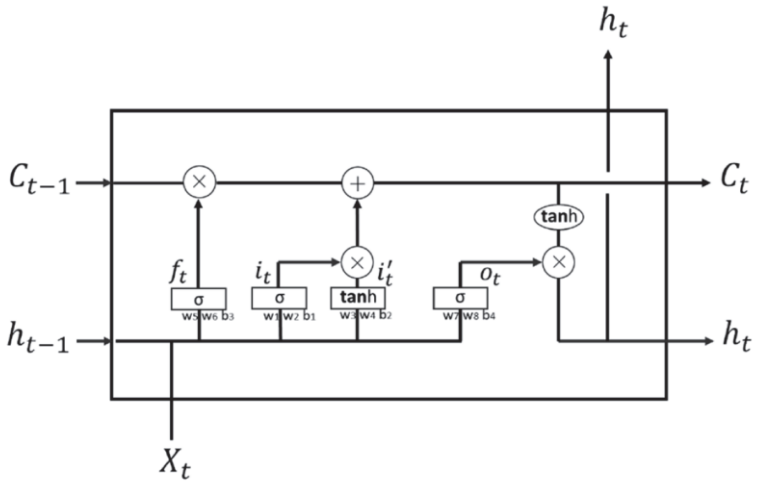

Figure 4. LSTM structure

The most widely used variant of this model is BiLSTM (Bidirectional LSTM). BiLSTM can be understood as two LSTM nodes that are trained simultaneously, and their hidden state output are merged at the same time t. This allows the networks to have both forward and backward information about the sequence at every time step. It can manage longer sequences and semantic meanings, therefore, it significantly outperforms unidirectional LSTM[8]. Other variants of this model are presented in the literature, Jie Ji et al. [9] compare versions of this architecture such as the Stack LSTM, the Fully-Connected LSTM, and the Bidirectional Fully-Connected LSTM.

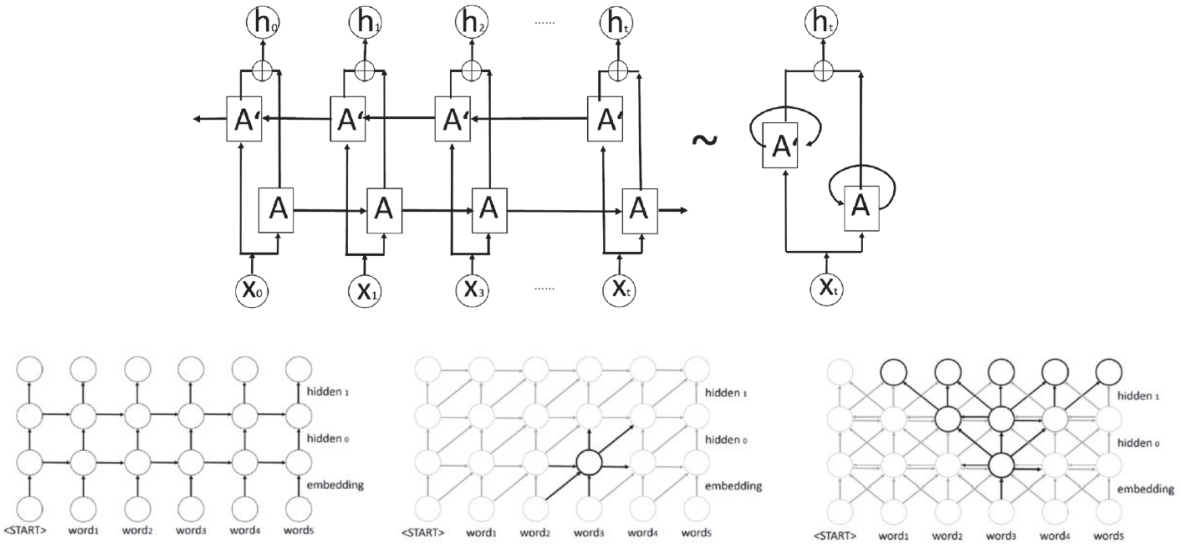

Figure 5. LSTM Variations

2.3.2 Conditional-Random-Field (CRF)

As mentioned in previous paragraphs, in the literature of the NER problem, when the BiLSTM model is used to build a hidden vector that is able to represent the context of both directions of each input token, the most commonly used to

perform the final classification of each token to identify the entities of the sentence is to use the statistical model of Conditional Random Fields (CRF) [10]. The output of the BiLSTM is then fed to a linear chain CRF, which can generate predictions using this improved context [11]. This combination of CRF and BiLSTM is often referred to as a BiLSTM-CRF model.

The BIO / IOB format (short for inside, outside, beginning) is a common tagging format for tagging tokens in a chunking task in computational linguistics. This scheme was initially proposed by Ramshaw and Marcus (1995) [12], and the meaning of the IOB tags is as follows:

- The I-prefix indicates that the tag is inside a chunk (i.e. a noun group, a verb group etc.).
- The O-prefix indicates that the token belongs to no chunk.
- The B-prefix indicates that the tag is at the beginning of a chunk that follows another chunk without O tags between the two chunks.

An example of the use of this BiLSTM-CRF architecture used for BioNER is explained in the paper by Raghavendra Chalapathy et al. [8].

2.3.3 Graph Convolutional Network (GCN)

Graph Convolutional Networks (GCNs) are a class of neural networks that operate directly on graphs, allowing complex relationships and dependencies between data to be modelled in ways that traditional neural networks cannot. Unlike conventional networks that operate on data structured as grids (such as images), GCNs deal with data structured as graphs, where nodes represent entities and edges represent their relationships.

Although GCNs are less conventional in natural language processing tasks such as NER, their ability to model complex relationships makes them useful in specific variants of these problems. For example, in a graph where nodes represent entities in a text and edges their relationships, a GCN can help identify and classify entities (genes, proteins, etc.) based not only on textual information but also on the structure of their relationships. In the paper by Yinxia Lou et al. [13] we can see how the use of this type of networks in combination with BiLSTM allows to perform dictionary-based BioNER effectively.

2.3.4 Pre-trained Language Models

The emergence of large, pre-trained language models (PLMs) has enabled a breakthrough in the NLP field in recent years. For many NLP tasks, the use of PLMs has outperformed the state-of-the-art results. The key idea of the PLMs is to learn a generic, latent representation of language from a generic task once, then share it across different NLP tasks. Language modelling serves as the generic task, with abundant self-supervised text available for extensive training [14].

Before its appearance, most studies focused on creating specialized models for specific tasks that were not extrapolable to other types of NLP tasks. This has led to the emergence of the “pre-training then fine-tuning” NLP paradigm [15]. This paradigm allows the exploitation of unlabelled data for training the base models and then fine-tuning them with a reduced set of labeled data. Despite having limited labeled data, the effectiveness of subsequent natural language processing tasks is significantly improved.

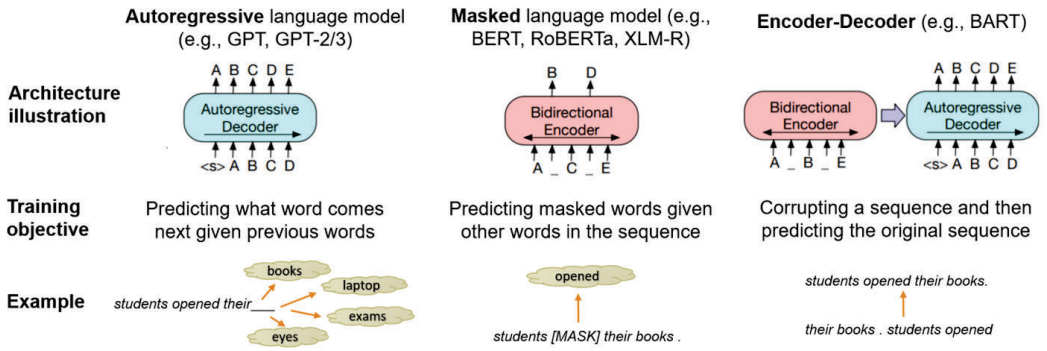

Figure 6. Pre-trained Language Models examples

There are three classes of PLM architecture and three classes of training objective:

- decoder-only (GPT, Gopher)
- encoder-only (BERT, XLM-R)
- encoder decoder (BART, T5, T0)

Moreover, models can be trained autoregressively (predict the next word based on the left-hand context), on masked language modelling (fill in the blank given context on both sides), or on a range of denoising tasks where the model must undo some corruption of the original sequence, such as sentence permutation, token deletion, or span deletion [14].

2.3.4.1 BERT (encoder-only)

BERT [16], which stands for Bidirectional Encoder Representations from Transformers, is designed to pretrain deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers. Consequently, it is possible to fine-tune the pre-trained BERT model by adding only one additional output layer to develop state-of-the-art models for various tasks, such as question answering and linguistic inference, without significant modifications to the task-specific architecture.

BERT overcomes the unidirectionality limitation of previously developed models by using a "masked language model" (MLM) pre-training objective. The authors used two different tasks to pretrain the language model:

- The model randomly masks some of the input tokens, with the goal of predicting the original identifier in the vocabulary of the hidden word, based only on its context (also referred as Cloze Task in the literature). Unlike the pre-training of left-to-right language models, the MLM target allows combining left and right context, facilitating the pre-training of a bidirectional deep transformer.

- Next Sentence Prediction (NSP) helps BERT learn about relationships between sentences by predicting if a given sentence follows the previous one. Specifically, when choosing the sentences A and B for each pre-training example, 50% of the time B is the actual next sentence that follows A (labeled as IsNext), and 50% of the time it is a random sentence from the corpus (labeled as NotNext).

The input consists of a sequence of tokens, on which each layer executes self-attention, processes the results through a feed-forward network and then transfers them to the next encoder. As a result, a vector is generated for each input token with a hidden size. Other language models like Glove2Vec [17] and Word2Vec [18] have built context-free word embeddings whereas BERT will provide context. In the original paper [16] the authors fine tune the model to perform several tasks using datasets such as SQuAD [19] and GLUE [20].

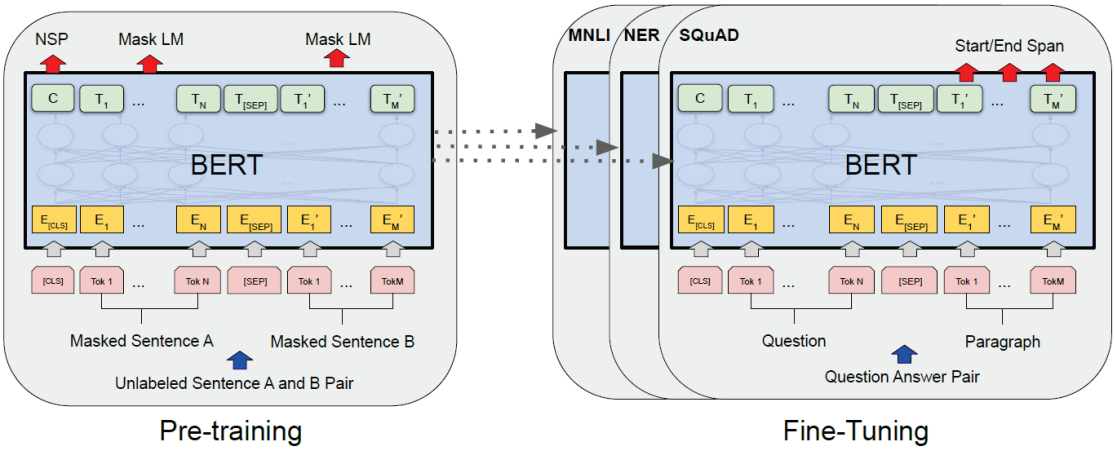

Figure 7. BERT Pre-Training and Fine-Tuning

Due to the good results obtained with BERT, variants of this model have emerged, retrained to improve performance in specific domains.

BioBERT [21] (Biomedical BERT), is a domain-specific language representation model pre-trained on large-scale biomedical corpora. With almost the same architecture across tasks, BioBERT largely outperforms BERT and previous state-of-the-art models in a variety of biomedical text mining tasks when pre-trained on biomedical corpora. While BERT obtains performance comparable to that of previous state-of-the-art models, BioBERT significantly outperforms them on the following three representative biomedical text mining tasks: biomedical named entity recognition, biomedical relation extraction and biomedical question answering.

2.3.4.2 Autoregressive Transformers (decoder-only)

Decoder-only or Autoregressive transformers, in contrast to the only-encoder models that are specialized in analyzing and interpreting text, are designed to generate new text. These models predict each subsequent token based only on the previously generated tokens, this means that they are unidirectional, the models learn to predict each word only based on the information available up to that point in the sequence.

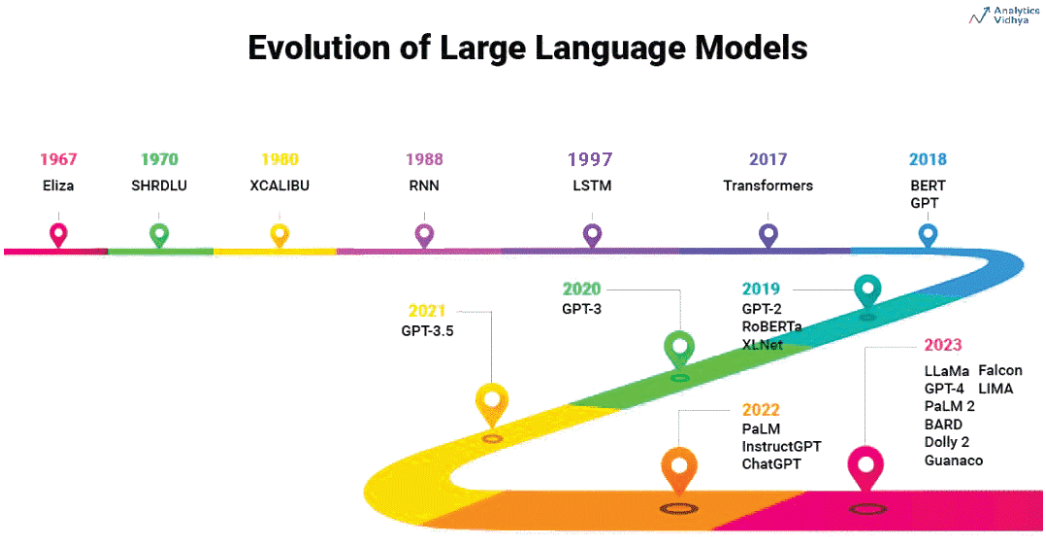

Figure 8. Evolution of LLMs [22]

Most of the big tech companies have launched autoregressive models, the most important of which stand out:

- **OpenAI**: in the last few years, OpenAI has released three main autoregressive models, known as GPTs: GPT2, GPT3 and GPT4, with some variants of each one, being GPT-4o [23] the most powerful, launched in May 2024. These models have significantly pushed the boundaries of natural language generation, enabling applications ranging from simple text completion to complex tasks like code generation and creative writing. Of the GPT models the only one that is available for use and download to all users is GPT2, the other models are only available through the ChatGPT website [24] for free including GPT4. GPT4-o can only be accessed by subscription.

- **Meta**: Among the models that Meta has developed, the most powerful to date is Llama 3 [25], launched in April 2024. It builds upon the foundation laid by LLaMA 2, featuring models with 8 billion and 70 billion parameters, and offers substantial improvements in both capabilities and performance. LLaMA 3 has been optimized for various applications, including text generation, chatbot integration, and image creation. LLaMA 3 is free and open source, Meta has made it available to the research and development community free of charge and can be found at Huggin Face.

- **Google**: Some of Google's most significant contributions to the language model landscape have been PaLM and PaLM 2, but above them is Gemini [26] launched in December 2023. Like ChatGPT, Gemini model can be used via chat through their website and is not available for download. To use models such as Gemini, developers and enterprises generally must access them through cloud services offered by Google, such as Google Cloud AI, where they can integrate the model's capabilities into their own applications and services, paying for usage on a pay-as-you-go basis. Google has additionally released Gemma as a family of open-source LLMs, providing free access and the ability to download the models. Gemma models are available in two main sizes: 2B and 7B parameters, and are designed to be lightweight and efficient, based on the same technology as the Gemini models.

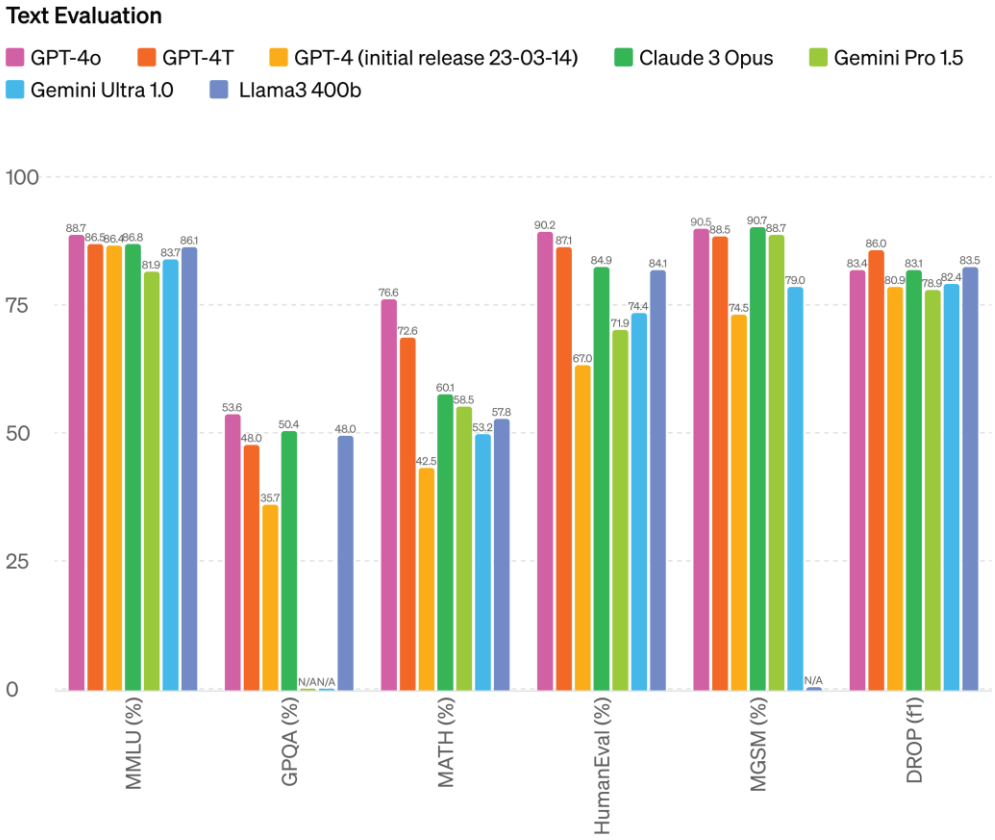

Figure 9. Comparison between different Autoregressive Models

Of the previously discussed models, the most recent is GPT4-o, the graph in Figure 9 shows the comparison published by OpenAI at the launch of this model, where for the Text Evaluation task it managed to reach the quality of most models for different benchmarks.

2.4 Training Approaches

As mentioned in previous sections, PLMs are models that have learned general language representations through unsupervised training of a large set of unlabelled texts. These PLMs can be used for different tasks that will require a smaller specialized dataset, and depending on the model and the task for which we want to use it, there are different approaches to adapt it.

2.4.1 Feature Generation

Feature generation involves using the representations learned by the PLM during pre-training to enhance the performance of downstream tasks, using these representations of the input text as the input for other machine learning models. For example, in text classification, the PLM can transform raw text into high-dimensional vectors that capture semantic information, which can then be fed into a classifier like a support vector machine or a logistic regression model or a neural network.

It is a great option when there is limited labeled data for the downstream task, as the pre-trained model provides a feature set that encapsulates a wide range of linguistic patterns. Feature generation is often used in scenarios where fine-tuning a large model is computationally expensive or unnecessary.

2.4.2 Fine-Tuning

Fine-tuning is one of the most used approaches to adapt pre-trained language models to specific tasks, where the model is further trained on a task-specific dataset, which typically involves supervised learning with labeled data. Fine-tuning involves adjusting the weights of the entire model or a subset of layers to minimize the loss.

There are several strategies for fine-tuning:

- **Full fine-tuning**: All parameters of the PLM are retrained based on the new task-specific data. This approach can achieve high performance but requires substantial computational resources.
- **Partial fine-tuning**: Only a subset of layers (usually the top layers) is fine-tuned. This can be more efficient while and still leveraging the powerful representations of the PLM.
- **Adapter modules**: Small neural networks (adapters) are inserted into the pre-trained model layers and trained on the new task. This maintains the original model weights and only trains the adapter parameters, reducing computational costs and retaining the generality of the original PLM.

2.4.3 Zero-Shot Learning

Zero-shot learning is an approach where a model, at test time, performs tasks it has never seen before during training. This capability comes from the model's understanding of the association between different tasks and the knowledge it has learned during pre-training, where it leverages its general language knowledge to make predictions for new tasks based on provided prompts.

For example, our model is given a description of the classes and a new text to classify, then the model generates predictions by comparing the new text against the class descriptions, without having been fine-tuned on specific examples of those classes.

Zero-shot learning is particularly valuable in scenarios where obtaining labeled data is challenging or impossible, it demonstrates the versatility of PLMs to generalize across various tasks with minimal additional input. Recent models (such as GPT) have shown impressive zero-shot performance across a wide range of NLP tasks, making this approach a significant area of research.

2.5 Data Augmentation

Data augmentation in the machine learning context is a technique that consists of increasing the number of instances of our training data in order to achieve better results with our models. Data augmentation can make models generalize better, improve unbalanced datasets, minimize labeling efforts, and limit the amount of data needed. In text classification tasks this technique allows models to learn more linguistic patterns during the training phase, and improve the model robustness to variations.

In the paper by Markus Bayer et al. [27], a review is made of the taxonomy of data augmentation methods for text classification, illustrated in Figure 10.

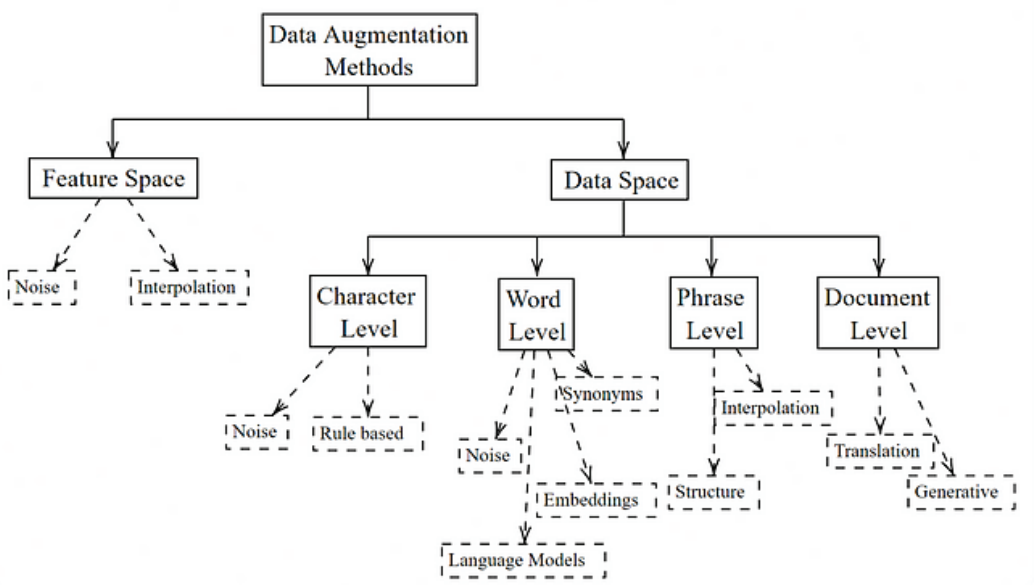

Figure 10. Taxonomy of different data augmentation methods.[27]

Word Level: This level of data augmentation involves the transformation of individual words in the text, such as changing them to synonyms, word shuffling, and other modifications that introduce variability but do not modify the meaning and context of the text.

Phrase Level: This level focuses on altering the structure and composition of sentences. It employs techniques such as paraphrasing, sentence shuffling or introducing grammatical variations. The objective is to diversify the data set with different formulations of ideas while maintaining the essence of the original content.

Document Level: This level involves the entire document or text to perform data augmentation. Changes made at this level are more substantial, involving insertion or deletion of entire paragraphs, reordering of sections and even changing the style of writing.

For our use case we will use data augmentation at the phrase level. This type of data augmentation methods consists of creating new training samples from existing ones by modifying the sentence structures. Some of the modifications that can be made to the sentences to obtain new training data are:

- **Structure transformation**: This type of transformations uses features and components of the sentence structure to generate the modified texts. This type of transformations uses features and components of the sentence structure to generate the modified texts. They can be based on grammatical dependencies such as part of speech (POS) relations. The semantics of natural language are sensitive to text order, while slight order change is still readable for humans, therefore, the random

swapping between words even sentences within a reasonable range can be used as a data augmentation method.

- **Round-trip translation**: is a method of producing paraphrases with the help of translation models. A phrase is translated into another language (forward translation) and then translated back into the source language (back translation).

- **Generative Methods**: as language generation capabilities have increased significantly, current models can create very diverse texts and can therefore incorporate new information. Generative methods to augment data consist of training language models to produce sentences similar to those of the training data.

2.6 Human Phenotype Ontology (HPO)

The Human Phenotype Ontology (HPO) is a bioinformatics resource used to standardize the description of human phenotypes observed in genetic diseases. It is a structured and unified language that allows the annotation and comparison of clinical data related to phenotypic manifestations in individuals affected by genetic diseases. The HPO organizes phenotype terms into a hierarchy, facilitating representation of the diversity of clinical features associated with specific genetic diseases.

The HPO was launched in 2008 to provide a comprehensive logical standard to describe and computationally analyse phenotypic abnormalities found in human disease. The HPO is now a worldwide standard for phenotype exchange. The HPO has grown steadily since its inception due to considerable contributions from clinical experts and researchers from a diverse range of disciplines [28].

The HPO consists of approximately 18,000 terms organized in an acyclic graph and connected by simple “is-a” (subclass of) relationships [29]. These relationships are transitive, meaning that they are inherited in all paths to the root. Relationships are transitive, meaning that they are inherited in all paths to the root. “Phenotypic abnormality” is the main subontology of the HPO and contains descriptions of clinical abnormalities. Other subontologies are included to codify inheritance patterns, clinical course, and modifiers of abnormalities [29].

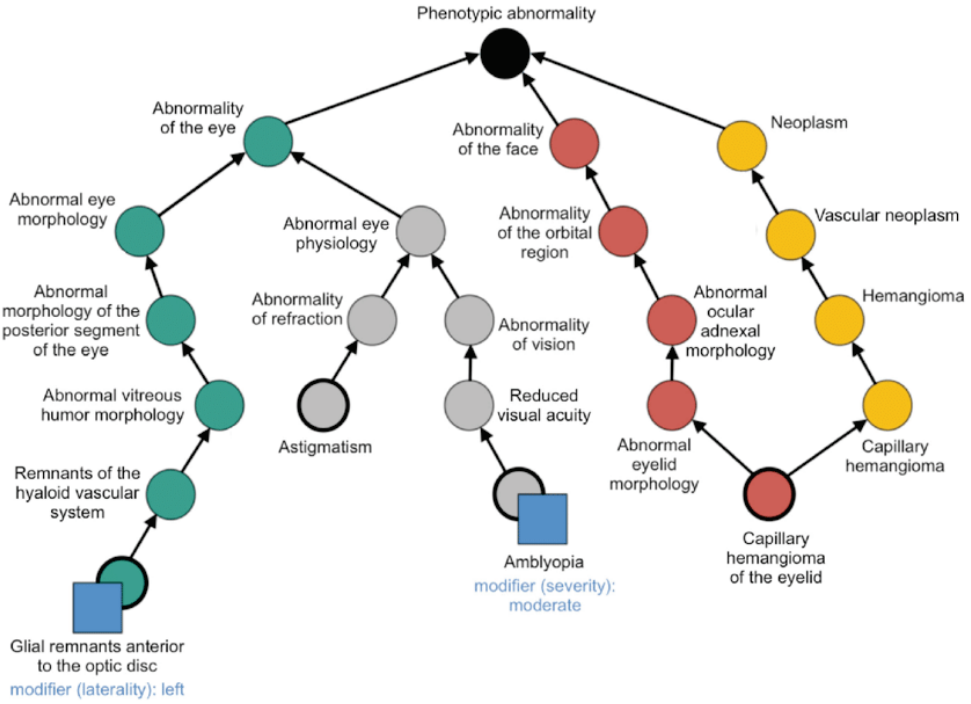

Figure 11. Example of the HPO hierarchy [30]

2.7 Related Work

In the literature we can find different works that have attempted to solve the problem addressed in this thesis. The aim of this section is to list the advances that different publications have made about NER of HPO terms. It should be noted that all the work previously done is framed in the language of English, which makes the challenge addressed in this thesis greater since the adaptation to Spanish is a key point.

In 2021, Ling Luo et al. [31] proposed in their paper the method called "PhenoTagger" which presented a hybrid approach for the recognition of HPO concepts in medical texts, combining dictionary-based and machine learning methods. The dictionary they build that is based on HPO uses all the concepts and synonyms available in this ontology. It uses the dictionary of HPO and biomedical literature available in PubMed Central to build a remotely supervised training dataset, which avoids the need for a large corpus of manually annotated data. The trained model is specifically a BioBERT that classifies each input sentence into a corresponding concept label. It combines the results of the dictionary-based method and the machine learning-based method to improve the performance in identifying phenotypic concepts, also considering overlapping concepts.

A couple of years later, in 2023, Yuhao Feng et al. [32] published in their paper their method called "PhenoBERT". It's a deep learning model that combines a first module consisting of two hierarchical convolutional neural networks (TLH-CNNs) that pre-selects a short list of candidates HPO terms that are passed to a second core module consisting of BERT that evaluates whether those text

segments correspond to any HPO term. Additionally, BERT can assign an ancestor HPO term to a text segment when it is not possible to recognize the direct HPO term, mimicking the term assignment process by a human. PhenoBERT is significantly faster than PhenoTagger due to the use of the TLH-CNNs module as a pre-screening step, and it shows better accuracy and recall.

With the recent advancement of PLLMs in recent years, Jingye Yang et al. [33] present in their paper 2 models, one is PhenoBCBERT which is based on BERT similar to what we saw in the previous articles but this model is pre-trained with Bio+Clinical BERT, and another one called PhenoGPT based on OpenAI GPT models. The GPT model is used with both a prompt-based learning and fine-tuning approach to improve recognition of phenotypes, including those not represented in the standard HPO vocabulary.

2.8 Evaluation Metrics

In the development and evaluation of machine learning models in classification tasks, evaluation metrics play a key role. These metrics allow us to quantify the capability and effectiveness of a model in generating accurate predictions on test data. The choice of the most appropriate metric for each case depends on the specific context of the model application. For example, in a medical environment the consequences of a false negative (failure to detect a disease when it is actually present) is more critical than a false positive. This is why understanding and selecting the most appropriate assessment metric is essential to ensure that the model is practical in real-world and technically competent.

Precision is a metric that measures the accuracy of the model's positive predictions, it reflects what percentage of the instances that the model predicted as positive are actually positive. Precision is particularly useful in situations where the number of false positives is high.

$$Precision = \frac{TP}{TP + FP}$$

Recall, also known as sensitivity, measures the model's ability to identify positives. It calculates the percentage of predicted true positives among the set of true positives and false negatives. This metric becomes very important in contexts where it is important to capture all positive cases, for example, when trying to detect a serious disease in a patient, a model with high recall will ensure the detection of most positive cases, reducing the risk of not treating a sick patient.

$$Recall = \frac{TP}{TP + FN}$$

When is required to have a balance between accuracy and recall it is very useful to use the **F1 Score** metric. This metric is calculated as the harmonic mean of precision and recall. Unlike the simple mean, the harmonic mean penalizes more the outliers so F1 is a more robust measure when you want to balance precision and recall. The F1 Score is especially relevant in scenarios where both false positives and false positives have significant consequences.

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

3 Methodology

3.1 Problem definition

With the emergence of electronic medical records, more and more efforts are being made to include as much information as possible in these digital records. It is in this context that the need arises to use clinical terminologies which play a vital role in the semantic interoperability of information. The use of clinical terminologies and ontologies, in addition to making a great contribution to the interoperability of information between electronic systems, also plays an important role in clinical studies and processes.

Correct coding of medical phenotypes through clinical ontologies such as HPO facilitates the identification of patterns and correlation of symptoms with possible diseases. This can speed diagnosis and improve accuracy, especially in rare and complex diseases where phenotypes are key to correct identification.

The problem in which this project is framed lies in the fact that most of the phenotype information recorded in the electronic medical record is in free text format, which has led many institutions and organizations to make many efforts to transform this information into a standardized format. When designing an AI model that attempts to automate this type of standardization process, a series of problems and challenges must be considered.

One of the first problems to be addressed is the lack of labeled data to train a model that is capable of labeling phenotypes in text. This means that the only resource we can use as a data source is the HPO itself. The complication of using only this ontology as a data source is the classification of synonyms that are not included in it. There is a difficulty in being able to identify ways of expressing phenotypes that have not been seen previously, and this is where the use of deep learning has an important role to play in achieving greater generalization of terms.

Specifically, the amount of data available for the Spanish case is even smaller, since the official HPO itself is not completely translated into Spanish, only the main tags, but not the synonyms or descriptions. Small datasets with labeled examples of HPO phenotypes can be found in the literature and are often used to test the models, but they are all in English.

In the state of the art reviewed in previous sections, we have seen that there are some articles focused on addressing this task of HPO terms recognition, but

again all works do it in the context of English, so there is no history of standardization of HPO phenotypes in Spanish with NLP techniques.

This section explains the steps followed for the creation of a hybrid NLP model that manages to solve, to a certain extent, all these challenges.

3.2 Libraries, Languages and environments

The programming language used for this project was Python [34] which is one of the most widely used programming languages in the world, and specifically in the context of artificial intelligence, due to its compatibility with model development frameworks such as TensorFlow or Pytorch. The use of this programming language allows the import of libraries for the use of different functions:

- **Tensorflow** [35]: is an open-source library developed by Google for numerical computing and machine learning. It's especially useful for building and training deep learning models and neural networks. TensorFlow provides a wide range of tools, libraries and community resources to facilitate the creation of machine learning models.

- **Transformers** [36]: is a library developed by Hugging Face that provides pre-trained models for NLP. These models use transformers architectures, such as BERT, GPT and others, for tasks such as language translation, text summarization, text classification and more.

- **Numpy** [37]: is a fundamental library used for scientific programming in Python, and in particular for programming in Data Science, engineering, mathematics or science. NumPy is essential for efficient numerical calculations and is the basis for many other scientific libraries in Python.

- **Matplotlib** [38]: is a data visualization library. It allows programmers to create high quality static, animated and interactive graphics in a variety of formats.

- **NLTK (Natural Language Toolkit)** [39]: is a complete library for NLP. It includes tools for working with text, grammatical tagging, parsing, etc.

- **spaCy** [40]: is a high-performance NLP library. It offers a wide variety of functions, including tokenization, part-of-speech tagging, parsing and lemmatization.

- **DeepL** [41]**:** is a library that provides tools for automatic translation of texts between different languages. It uses translation engines and pre-trained models based on transformer architecture to provide accurate and fast translations.

Virtual environments have been used for the development of the project through Anaconda [42] which is a distribution of both Python and R that simplifies the management and deployment of environments and packages. This allows users to keep different Python versions and packages isolated from each other, which is crucial for avoiding dependency conflicts.

3.3 Tagging Model Architecture

The system to be built uses the "Phenotagger" model explained on the state of the art as a reference. The designed system consists of two main modules on which the HPO phenotype annotation system will be based: Dictionary Matching and the deep learning model. Both modules use a dictionary based on the HPO ontology as a source of knowledge.

In the general architecture of the model, different processes can be distinguished:

1. Dictionary construction
2. Deep Learning Model Training
3. Input texts preprocessing
4. Dual tagging of the text: Deep Learning and Dictionary based methods
5. Result combination of both approaches

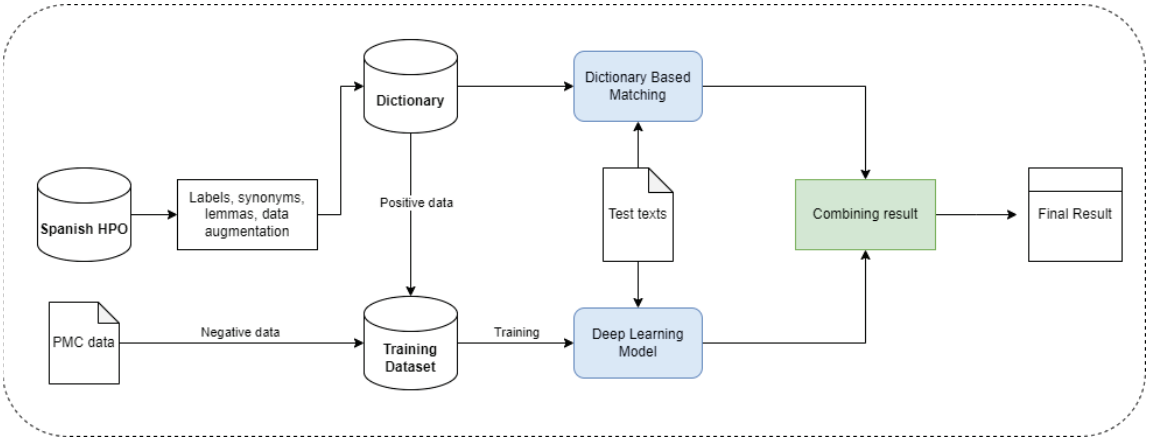

Figure 12. Tagger Architecture

The process can be divided into two phases, training and testing. In the training phase, first a preparation of the training data is done, which includes the construction of a dictionary containing the HPO concepts, and a training dataset composed of this dictionary and biomedical text from the literature. In the test phase, the HPO dictionary and the trained model are used for dictionary-based matching and a deep learning-based method is used to recognize HPO concepts from biomedical input texts. Finally, the results of the two tagers are combined with a set of logical rules.

3.4 Data Preparation

3.4.1 Dictionary

The dictionary that has been built has two main objectives; to serve as a source of concept search by means of string-matching techniques; and to be part of a training dataset for a deep learning model. The dictionary has been built using the HPO ontology explained in previous sections. As already mentioned, one of the limitations was that the official ontology published on the web is translated into several languages, but in the case of Spanish this translation only covers the labels of the terms, the descriptions and synonyms are in English.

For the tasks of translation and maintenance of this type of open-source ontologies, usually work teams from different institutions carry out their translations and then include them in the official ontology. Fortunately, this final master's thesis is framed in the context of a working group that is carrying out the last steps of a Spanish translation of the HPO in which numerous Spanish synonyms have been included to the ontology. This translated version of the HPO has been validated by experts in the field of genetics and is awaiting submission and acceptance for publication. Although not yet officialised, it will serve as a source of data for this project.

The HPO has a hierarchical structure of subclasses, there is a parent term for all other terms, which is “All” with the code HP:0000001. From this term, 7 subclasses of terms come out directly, which are the ones shown in Figure 13. For the construction of the dictionary, the subclass that will be of interest to us will be only “Phenotypic Abnormality”, which contains all the phenotypes that we want to identify in free text.

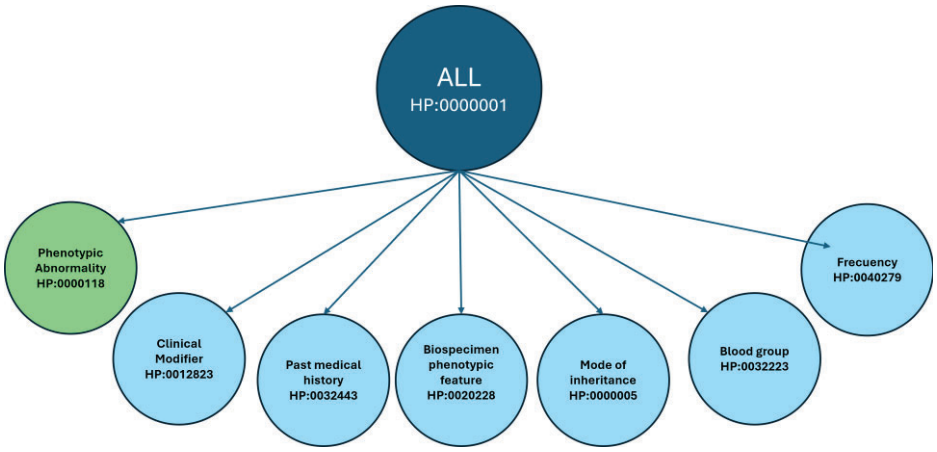

Figure 13. HPO Main subclasses of the term "All"

Within the Phenotypic Abnormality subclass, the ontology is divided into 23 other branches organized by the system or body area in which the phenotypes are manifested. The terms contained in these branches will be those of interest for inclusion in the dictionary. Taking only the terms that are subclass of phenotypic anomaly, we have a total of 17.957 codes to include in the dictionary. Figure 14 shows the number of terms in each of the 23 branches in more detail.

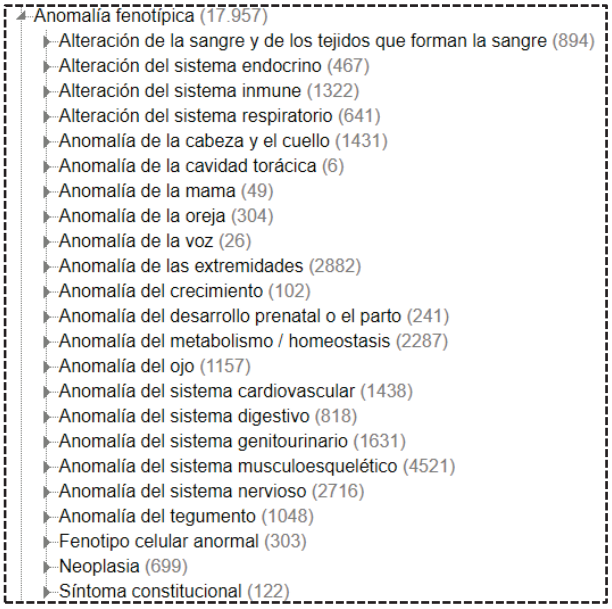

Figure 14. The 23 subclasses of Phenotypic Abnormality

We can build our dictionary with the main label and the synonyms included for each term. The number of synonyms for each term is highly variable; many of the HPO concepts have no synonyms, while others may have many of them. In order to appreciate the large difference in the number of synonyms of the terms, Figure 15 shows the distribution of the number of synonyms per term in the Phenotypic Abnormality branch.

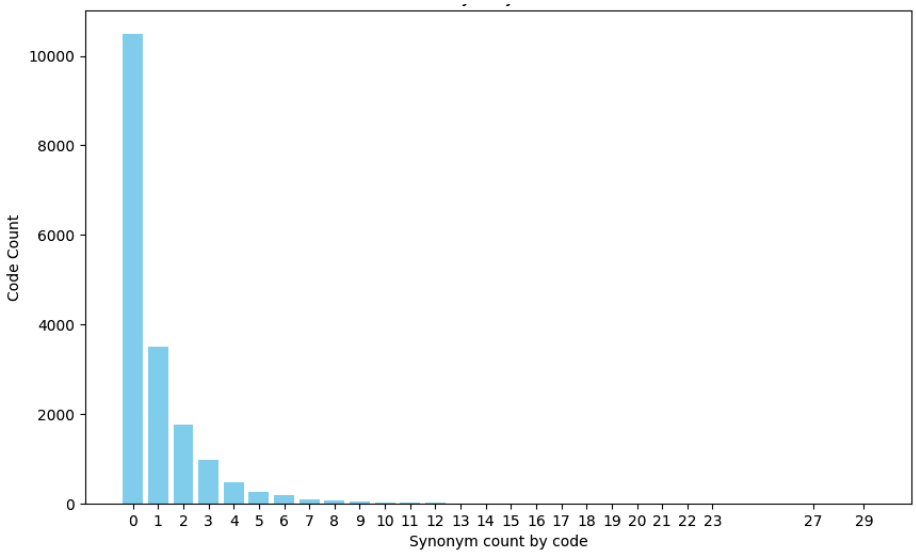

Figure 15. Synonyms Count distribution.

As can be seen, more than 10,000 terms have no synonyms, which accounts for more than half of the phenotypes. This will be part of the challenge that the model will have to overcome, to perform classification tasks relying on thousands of codes and with a small amount of training data for each class.

3.4.1.1 Lemmatization

In addition, with spaCy library we generated the lemmas of the concept names and synonyms to extend the dictionary, since it can obtain more general forms of the terms. Spacy library offers different pipelines in Spanish that, among other functions, provide a lemmatizer, and the one used for this task was "es_core_news_lg" [43]. Thus, in our dictionary we will not only have the labels and synonyms of the concepts but also the lemmatized versions of both, which in some cases will help to recognize a greater variability of the HPO concepts.

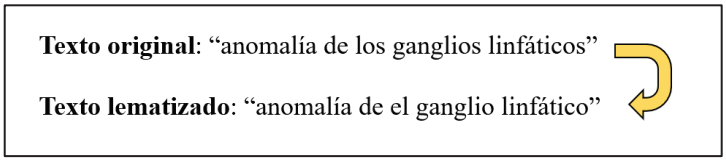

Figure 16. Lemmatization example

After applying the lemmatization of the dictionary, counting on the fact that we have filtered out those cases in which the lemmatized version of the phrase is equal to the original one, we obtain that we go from 37.268 instances in the original dictionary to a total of 55.769 instances.

3.5 Dictionary Matching

3.5.1 String Matching

To achieve an efficient dictionary lookup method, the Trie Tree Data Structure [44] is implemented to store the HPO dictionary. The Trie memory stores words using a tree structure where each node represents a character. A path from the root to a specific node represents a complete word or a part of it. Searching for words in a Trie is fast because the access time depends only on the length of the word and not on the total number of words stored.

Adding or updating words in a Trie is efficient, it does not require significant reordering or restructuring of the Trie. The Trie takes advantage of redundancies in stored words. For example, different words starting with the same prefix will share nodes in the Trie, which reduces the storage space required and increases search efficiency.

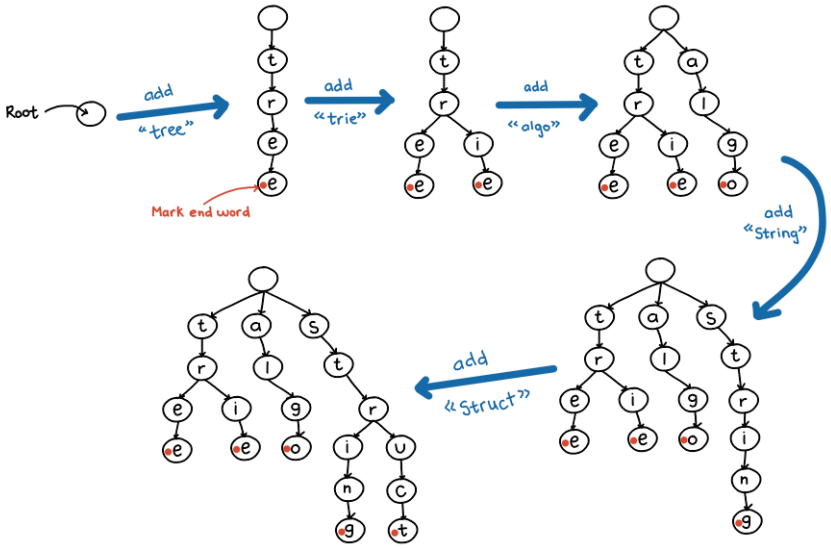

Figure 17. Trie Tree construction example [45]

Using the dictionary explained in the previous sections, the Trie Tree is built, which will contain in its nodes the labels, synonyms and lemmas of the HPO concepts. Then, the prefix search is applied for exact matching using the case-insensitive mode. The exact match dictionary-based method can achieve higher precision, since most of the concepts in the phenotype ontology are specific and unambiguous. However, its main problem is that the concept name usually has many synonyms with variant spellings, resulting in a lower retrieval rate.

Although some synonyms of the concept name are provided, it is impossible to cover all variations only by dictionary matching. The method cannot recognize the unseen concept synonyms in the dictionary. For this reason, the following sections explain the development of a model based on deep learning to improve recall.

3.6 Deep Learning model

This part of the method consists in a deep learning approach where it converts the HPO concept recognition problem into a text classification problem in which each HPO code in the ontology is a class.

Given a sequence of input words $x = \{x_1, x_2, x_3, \dots, x_N\}$ the model classifies it into one of the HPO labels $y \in [1, L]$ where N is the length of the sequence and L the number of classes of the problem (the number of unique HPO codes of the dictionary).

The classification process includes three steps: Training dataset construction and training, preprocessing the input text, and concept recognition.

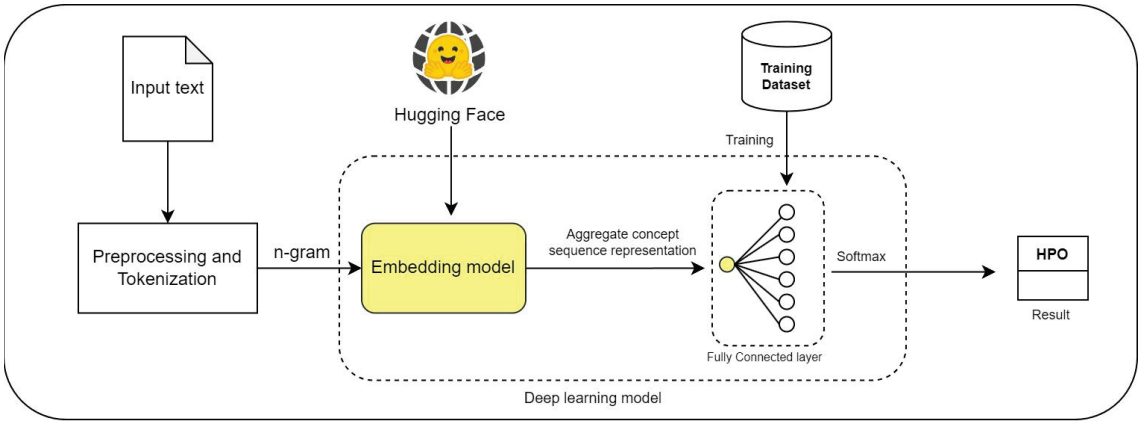

Figure 18. Deep Learning Model

3.6.1 Training dataset

In contrast to dictionary-based matching, what it's intended with the deep learning model is to recognize not only the phenotypes described in the dictionary but also to be able to recognize variants not included in it.

The training dataset has been created based on the dictionary explained in previous sections, in which has been taken from each HPO ID its label and synonyms (not the lemmas). Each of these terms corresponding to the same HPO ID will be our training instances (positive data) and the HPO ID corresponding to each instance will be the output of the model in the classification task.

On the other hand, for this classification task, we will also need an additional class in which to classify texts that are not phenotypes (negative data). This class will be assigned the code "HP:None" to indicate that the input text does not correspond to any of the HPO phenotypes. For this class we will also need training instances. To generate this training data, extracts from biomedical texts were taken from the literature, specifically from articles taken from PubMed.

The selected articles have been extracted from PubMed Central (PMC) [46] which is an open access digital repository of articles published in biomedical and life sciences journals. For the task of extracting texts from these articles, I have made use of the API developed by Donald C Comeau et al. [47] which convert these articles to BioC, a community-driven simple data structure in either XML or JSON format for sharing text and annotations.

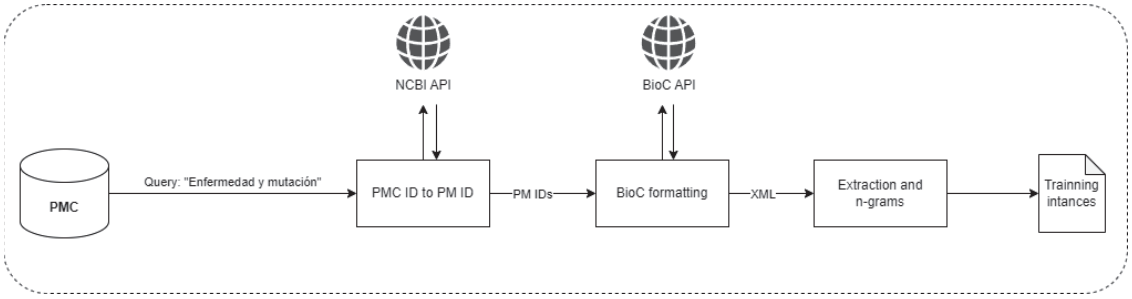

Figure 19. PCM articles extraction and formatting

The selected articles have been obtained through the query “enfermedad y mutación” in PubMed Central. In this way, the model can learn more useful information from the phenotype related texts than unrelated text, and phenotype often is associated with disease and mutation.

From the PubMed Central query, we obtain a list of 200 articles and their PMCIDs (PubMed Central Identifier). It is possible to use the API of Entrez Programming Utilities (E-utilities) [48] of the NCBI (National Centre for Biotechnology Information) to obtain the PMID (PubMed Identifier) corresponding to a PMCID.

This BioC API receives as input the PMIDs and returns the article in BioC format (XML) and being in this format the sentences of the paragraphs are extracted. From the obtained texts random n-grams are generated (where n can be from 1 to 10) and possible occurrences of dictionary phenotypes are filtered. Each of these n-grams will form a training instance labeled as “HP:None”.

A total of 37.268 positive instances and 35.000 negative instances were obtained once the training instances were gathered. Initially 35.000 negative instances were included to be balanced with the number of positive instances, but the impact of the number of negative instances on the test data is studied later in the results section.

3.6.2 Model

The trained deep learning model consists of two main parts. It has a first pre-trained contextualized word representation model which in all cases tested will be BERT-like models. The first token of this model is always a classification token “[CLS]” that is placed at the beginning of each input sequence. The resulting vector *C* of this token after the BERT encoding process is considered an aggregate representation of the whole sequence and will be the one we use for the classification task.

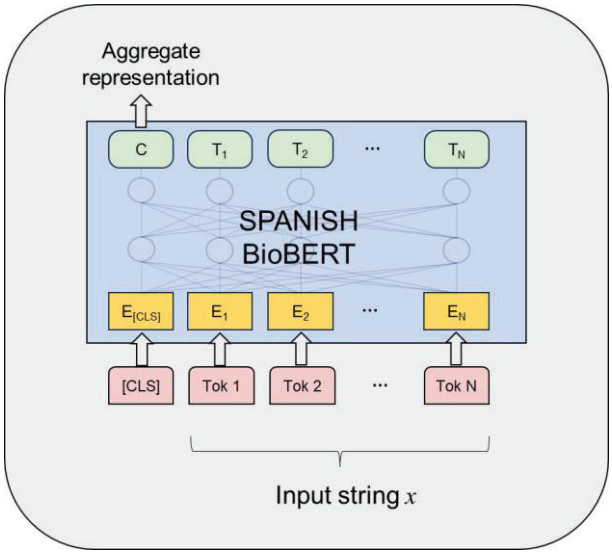

Figure 20. Spanish BioBERT

Table 1 summarizes the spanish BERT models we have selected as well as their characteristics, the first two models are RoBERTa models pretrained with biomedical data and developed by the Barcelona Supercomputing Centre (BSC) but with the difference that the second one has been additionally trained with texts extracted from EHRs. The third one is BERT a model developed by DCC UChile (Universidad de Chile) but not specifically trained with biomedical texts.

Model Name	**Hugging Face Name**	**Training Data**	**Parameters**	**Developer**
Biomedical language model for Spanish	PlanTL-GOB-ES/bsc-bio-es	The training corpus is composed of several biomedical corpora in Spanish, collected from publicly available corpora and crawlers.	124.643.328	Barcelona Supercomputing Centre
Biomedical-clinical language model for Spanish	PlanTL-GOB-ES/bsc-bio-ehr-es	The training corpus is composed of several biomedical corpora in Spanish, collected from publicly available corpora and crawlers, and a real-world clinical corpus collected from more than 278K clinical documents and notes.	124.643.328	Barcelona Supercomputing Centre
BETO: Spanish BERT	dccuchile/bert-base-spanish-wwm-uncased	The training corpus contains data from various sources, such as Wikipedia, ParaCrawl, EUBookshop, MultiUN, OpenSubtitles, and other multilingual resources.	109.850.880	DCC UChile - Universidad de Chile

Table 1. BERT Models

The aggregated representation of the input sentence that the model outputs is connected to a fully connected layer. The number of output neurons of this layer is L, which is the number of HPO codes in the dictionary.

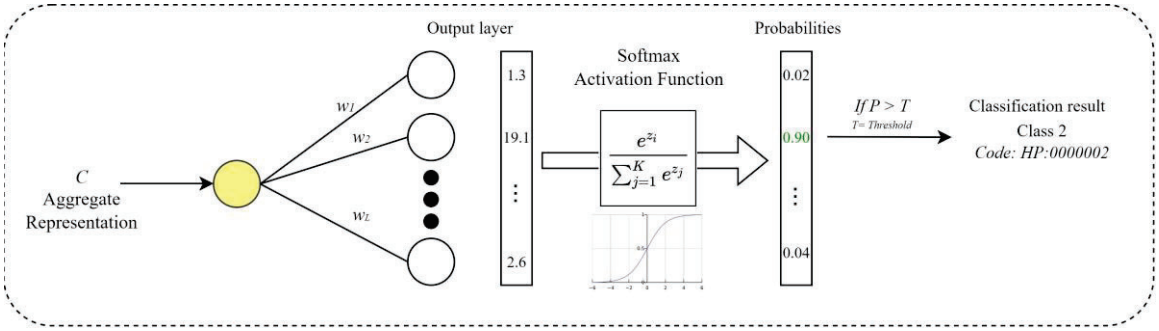

Figure 21. Fully Connected layer

This layer has a SoftMax function to translate the network output into a probability score matrix $P = softmax(CW^T + b)$ where:

- **P: Matrix of resulting probabilities**: Each element of P represents the probability that the input belongs to one of the possible classes.
- **C**: **Matrix of features of the input**: In a neural network, this matrix contains the values of the input features after passing through various layers of the network. In this case C will be a unidimensional vector with the representation value [CLS] of the string calculated by the previous transformer.
- **W: Weights matrix**: Connects the input neuron to the neurons of the output layer. This indicates that the weights are being multiplied by the input value in a form suitable for output computation.
- **b: Bias vector**: The bias is an additional value that is added to the weighted sum of the inputs to allow the model to better fit the data.
- **SoftMax trigger function**: This function converts the outputs of the linear layer ($CW^T + b$) to probabilities, ensuring that all probabilities sum to 1 and that each value is in the range [0, 1].

3.6.3 Training

For the training of the model, there were two ways to do it: Finne tuning of the whole model or opt for feature generation training only the dense layer. Due to the computational limitations of this project, we have chosen the option of using feature generation, in which only the final dense layer is trained to learn to classify the representations given by the BERT model. The BERT model selected to train the first version of the deep learning model was bsc-bio-es. The training runs were performed locally on a computer with 16GB RAM and a GPU NVIDIA GEFORCE RTX 3060.

Another limitation that has already been discussed in previous sections is the limited training data, which makes it impossible to make a division between training and validation data. To solve this, it has been decided to train the model and every 10 epochs test the model with the test dataset. This is the only way to see the real evolution of the training, as the accuracy of the training data does not provide information on whether overfitting is occurring.

As a first approximation we performed a training of 80 epochs and visualized the evolution of the training accuracy and loss in order to get an idea of how many epochs we should test. Figure 22 shows a rapid evolution of accuracy and loss, converging to a value of 0.8 and evolving very slowly thereafter.

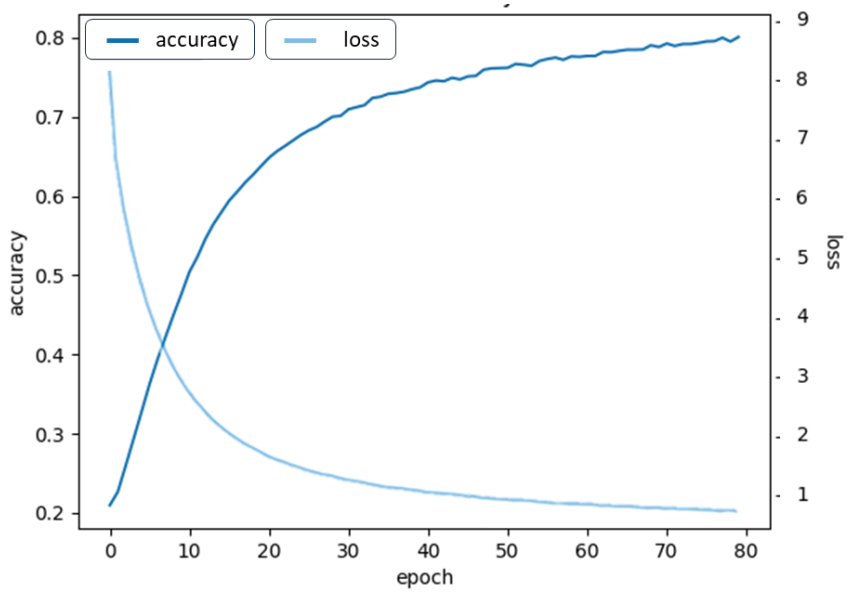

Figure 22. Training accuracy and loss evolution

Therefore, the parameters that have been chosen for the training are the following:

- Train it for 300 epochs to see the evolution of the metrics and see which is the most optimal value.
- The Adam optimizer with a learning rate of 10^{-4} has been used.
- The number of negative instances for these tests was 10.000 initially; the optimal number of negative instances is discussed in more detail later in the results section.

Every 10 training epochs, the resulting model at that time is used to label the texts of the test dataset. Accuracy, recall and F1 score metrics are obtained, and two different tests have been performed, one using only the deep learning model and the other using the combination of deep learning and the dictionary.

Figure 23 shows the evolution of the test results for the deep learning model during the training epochs, as the recall increases and the precision decreases but slowly, which causes a gradual increase of the F1 score. As it is a combined system, the intention is that the model tries to generalize as much as possible without overfitting. Overfitting the model would mean that it would be acting as a dictionary, learning exactly each of the instances, which we want to avoid.

Therefore, to see if the model is able to provide extra predictions that the dictionary is not able to do, the best alternative is to study what results it obtains in a combined way. In Figure 24 we can see that at epoch 0 the deep learning model does not provide any prediction, so those values of the metrics would correspond to the dictionary only. We can see how as the model is trained it causes the metrics to improve significantly, sacrificing a little precision but greatly increasing recall. At a certain point, although the recall continues to increase, the accuracy drops too much causing the F1 score to tend to decrease again, so it can be concluded that the time when the model performs best with the test data is around 110.

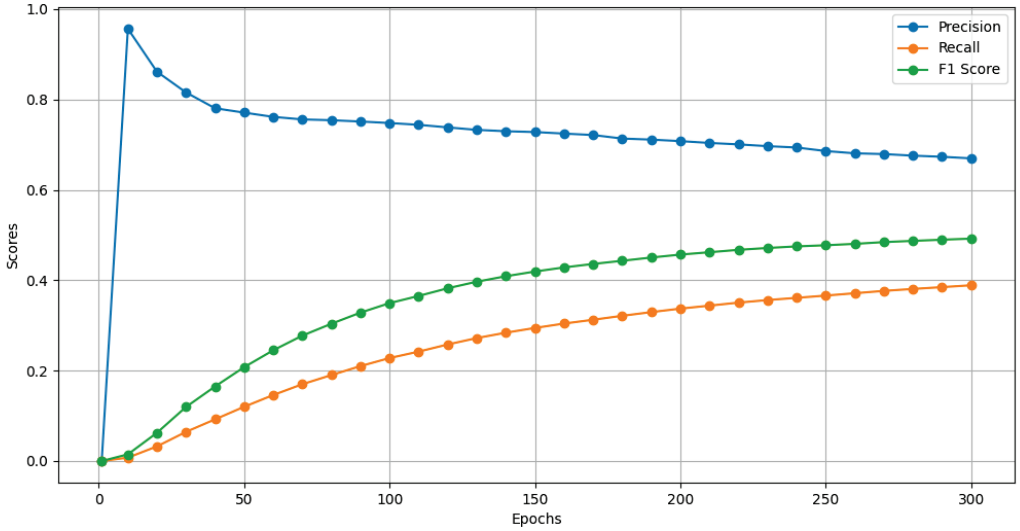

Figure 23. Test results evolution for Deep Learning model

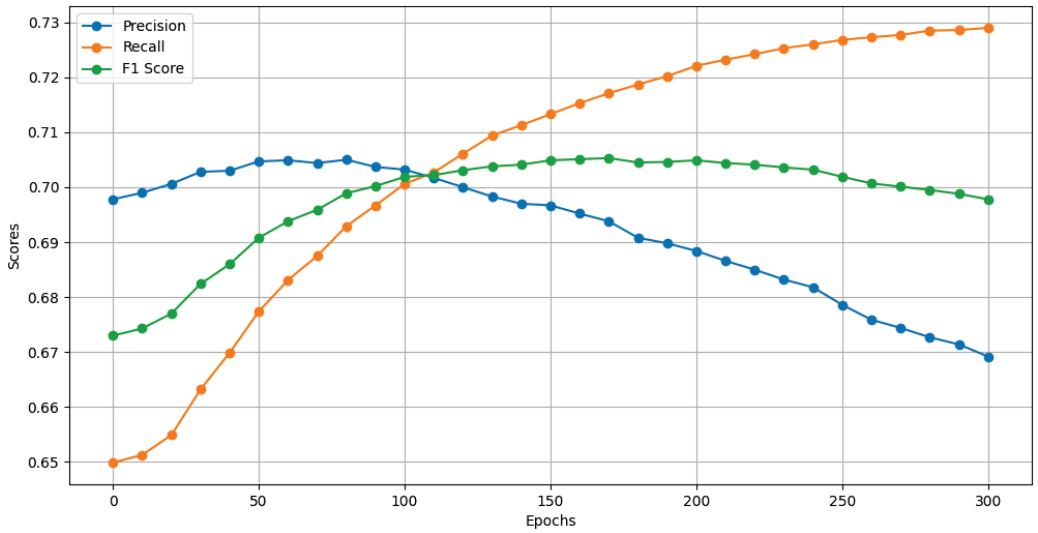

Figure 24. Test results evolution for Deep Learning model and dictionary method combined

3.7 Concept recognition process

In the previous sections we have reviewed the entire data preparation and model training phase. Once this phase is completed, the tagger can receive input texts that must first be pre-processed.

First, the input text is split into its different sentences, tokenized by words and labeling the part-of-speech of each one using the same spaCy model explained in Section 3.4.1.1 , and converted to lowercase. With these tokens, all n-grams are generated as the concept candidates, where $n \in [2,10]$. Both individual tokens and n-grams are sent to the dictionary-based method, but only n-grams are sent to the deep learning model, since unigrams will be recognized as false instances with high probability due to the little information they provide.

The reason for choosing 10 as the maximum length of the n-grams is that by reviewing the HPO we realize that less than 1% of the concepts in the HPO have a length greater than 10, so to make this method more efficient it has been decided to include only up to 10 as the maximum length of the n-grams. Another way included to make this process more efficient is to use the previously identified POS as a filter, so that if a candidate concept begins or ends with punctuation marks, prepositions, subordinating conjunctions, coordinating conjunctions or determiners then it is discarded.

The candidate concepts will go through both the dictionary-based method and the deep learning model. In the case of the deep learning model, it will return a probability P of the predicted HPO, which if greater than a threshold T will be accepted. The optimal value of T is studied in the results section.

3.8 Combining results

After the two annotation processes of the dictionary technique and the deep learning model, we will obtain two sets of detected concepts. We must merge the two sets of concepts to give a single set of HPO concepts as the final result.

First, we will assign a score of 1 to each of the concept obtained by the dictionary technique and the score of the results of the deep learning model will be the probability obtained after applying the SoftMax on the last layer. The following four rules are considered to consider the overlapping concepts between the two methods:

1. All non-overlapping concepts are maintained.
2. If the overlapping concepts have the same start and end positions in text but are mapped into different HPO codes, the concept with the highest score is retained.
3. If two concepts overlaps but both have the same HPO code, the concept with the highest score is retained.

4. If two overlapping concepts with different start and/or end positions in the text and different HPO codes, all the overlapping concepts are retained

3.9 Data Augmentation

To increase the number of instances of each of the dictionary terms, it has been decided to use data augmentation techniques. As mentioned in the state of the art, we are working at the phrase level, and we have decided to work with the Round-Trip translation technique, in which we will translate each of the original sentences from Spanish to English and back from English to Spanish so that in many cases new sentences with some variabilities are generated as shown in the following example:

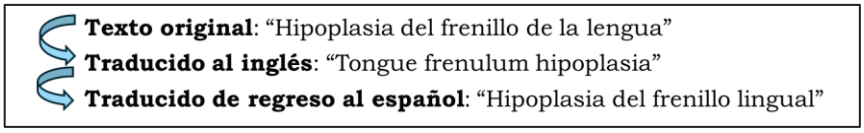

Figure 25. Round-Trip translation example.

Having to do thousands of translations, the best option has been to use a Huggin-face translation model to translate all terms in the local GPU, since libraries such as Translator or DeepL work via API and have a usage limit. With the models '*Helsinki-NLP/opus-mt-es-en'* [49] and '*Helsinki-NLP/opus-mt-tc-big-en-es*' [50] a Round-Trip translation of all the tags and synonyms of the ontology has been made and those cases in which this technique did not result in a sentence different from the original one have been excluded in order not to have repeated instances.

For each new synonym generated with the translation, a check is also made to ensure that it does not coincide with any instance of any other HPO concept. Sometimes, if the translator does not make a good translation of the concepts, it can give unexpected results that coincide with the name or synonym of another concept, generating noise when training the models if it is not eliminated.

After the execution of the data augmentation process, a total of 17191 new instances were obtained in the dictionary. Figure 26 shows the new distribution of synonyms by code after data augmentation. If we compare the distribution before and after data augmentation, it can be seen that we have managed to halve the number of HPO concepts that have 0 synonyms, and we have generally increased the number of synonyms per concept.

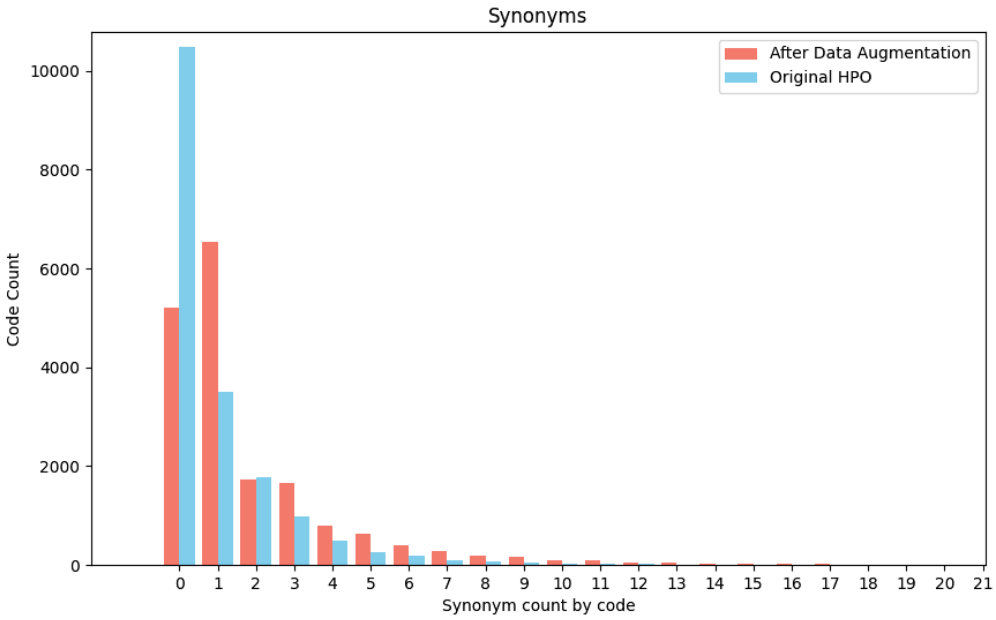

Figure 26. Synonyms distribution after Data Augmentation

The dictionary after data augmentation and subsequent lemmatization of all names and synonyms (explained in Section 3.4.1.1), had 82.791instances, which is 27,022 instances more than the lemmatized dictionary without data augmentation. The impact and effectiveness of this data augmentation technique will be studied in the results section.

3.10 Test Dataset

Finding a labeled data set to test the designed model was not an easy task. As mentioned in the previous sections, one of the main problems faced by this project was the lack of resources in Spanish, which meant that we had to resort to translation techniques for some of the development phases.

The dataset used to perform the tests was the Gold Standard Corpora (GSC) constructed by Tudor Groza et al. [51], which consists of an annotated HPO corpus composed of 228 abstracts manually annotated by 3 experts with a total of 1933 annotated HPO entities, covering 460 unique HPO concepts. These 228 abstracts were manually selected to cover 44 complex dysmorphology syndromes. On the other hand, Manuel Lobo et al. [52] identified some unlabelled entities and created an expanded version of the GSC (GSC+) by adding 881 entities and modifying 4 entities. Taking this benchmark as a reference, some modifications were made to adapt it to our problem:

- First, we note that some of the HPO entities annotated in the original texts belonged to HPO branches outside the “Phenotypic Abnormality” branch (HP:0000118), and as discussed in Section 3.4.1 the constructed

dictionary only contains this branch. Therefore, all GSC+ entities that did not belong to the Phenotypic Abnormality branch were filtered out.

- All abstracts have been translated into Spanish with the DeepL python library. Since the length of the abstracts was not large, we were able to use the free version of this translator engine to ensure the best possible translation.

After this filtering process, a total of 1471 HPO entities were obtained in the test texts, with 386 unique HPOs. Table 2 shows some of the most repeated HPO concepts in the test texts.

HPO concept	**Test instances**
Neoplasm (HP:0002664)	95
Basal cell carcinoma (HP:0002671)	79
Seizure (HP:0001250)	51
Vestibular schwannoma (HP:0009588)	38
Bilateral vestibular schwannoma (HP:0009589)	30
Branchial anomaly (HP:0009794)	27
Type C brachydactyly (HP:0009373)	26
Hearing impairment (HP:0000365)	25

Table 2. HPO concepts count of GSC+

4 Results and Discussion

This section aims to study the performance of the model on the test data set explained in the previous section, and to study the impact of different system design choices in obtaining results. The sections discussed in this section are:

- Which BERT model is best suited to this problem?
- How many negative instances should we use to train the model?
- Which threshold value is the most optimal?
- What impact does the data augmentation have on the result?
- Does using a lemmatized dictionary have an impact?
- Does the dictionary or the deep learning model work better?

Due to the complexity and characteristics of the test dataset, the metrics obtained from the tests performed have been calculated at the document level, only the set of concept ID labels within each document is considered, ignoring the exact concept positions in the text.

4.1 BERT models

The first comparison made was to use the different versions of BERT explained in Section 3.6.2 to see which of them performs a better representation of the language in this use case obtaining better results. The test performed was to train the 3 models with the same hyperparameters and the same training data. As studied in Section 3.6.3, the training parameters of the models are 110 training epochs, using 35.000 negative instances, with a batch size value of 32, and the Adam optimizer with learning rate of 10^{-4}.

The tests performed consisted of testing the entire system, combining deep learning and dictionary search to see the result obtained for de GSC+ dataset. The probability threshold applied to the deep learning model when making the predictions has been 0.90.

BERT MODEL	**Precision**	**Recall**	**F1**
bsc-bio-ehr-es	0.6182	0.7165	0.6637
bsc-bio-es	**0,7025**	**0.7512**	**0.7260**
bert-base-spanish-wwm-uncased	0.4619	0.6798	0.5501

Table 3. BERT models result comparation.

The results obtained are shown in Table 3, and unexpectedly, the model that obtained the best results was the *bsc-bio-es model*. This is surprising at first sight because one might expect the *bsc-bio-ehr-es* model to obtain better results since it is trained with more data than the previous one. But if we stop to analyse the differences between the two models, we can see that the extra texts with which this second model is trained are notes from the electronic medical record, which probably have linguistic expressions and phrases that can confuse the model.

The form of writing in biomedical literature will have a more structured and orderly style of expression than in clinical notes, which are often written in a rapid and shortened manner, introducing grammatical errors and incomplete expressions. This could explain the difference in the quality of learning linguistic representations of clinical words by the two models.

Finally, and as expected, BERT's model trained with generic texts not specialized in the biomedical field, was the one that obtained the worst results, since probably in the training texts there are many words of phenotypes that he has never seen, and he has not been able to generate good vector representations of these words.

4.2 Negative data impact

One of the points that can have the most impact when training our model is to study the number of negative instances to include in the training data set. As already mentioned in the methodology section, the negative instances are extracted from the biomedical literature so that the model learns to distinguish which parts of the text are part of a phenotype and which are not. The number of instances to be used during training is one of the parameters that had to be studied to find the most optimal value.

Initial intuition leads us to think that the more data the better, but this is not necessarily the case. In addition to the negative data, we have the positive instances, which as we have already studied, we have very few instances for each of the thousands of classes of the classification problem. The intention of this section is to run a series of tests with different versions of the model trained with a different number of negative instances. To see the evolution of the metrics, we have chosen to train models with the following data:

- **0 negative instances**: to study the classification capability of the model if it has never seen negative data.
- **10.000 negative instances**: to see the results when the model has more positive instances than negative ones. In this way there will be approximately the same amount of positive and negative data, thus balancing the dataset.

- **35.000 negative instances:** with this approach there will be approximately the same amount of positive and negative data, thus balancing the dataset.

- **50.000 negative instances**: To have more negatives than positives.

- **75.000 negative instances:** To have many more negatives than positives.

The training and test runs have been carried out using the combination of the dictionary system and the deep learning model with the same parameters used up to now, which, as has been studied, have proved to be the most optimal.

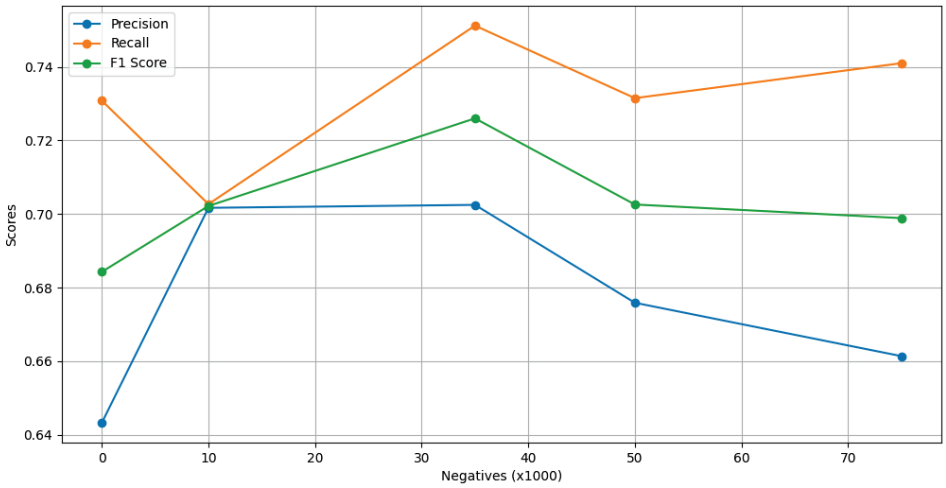

Figure 27. Negative instances impact on test results

Negative Instances	**Precision**	**Recall**	**F1 Score**
0	0.6433	0.7308	0.6843
10.000	0.7017	0.7027	0.7022
35.000	**0.7025**	**0.7512**	**0.7260**
50.000	0.6759	0.7315	0.7026
75.000	0.6614	0.7410	0.6989

Table 4. Negative instances impact on test results

Looking at the values of the metrics in Figure 27 and Table 4 for each of the models we can see how we obtain the best results when the model is trained with the same number of positives as negatives.

As the number of negative samples rises, the model's performance gradually enhances. The optimal performance, with an F1 Score of 0.726, is achieved when there are 50,000 negatives. However, introducing more negatives results in a slight decline in performance. An excess of negative samples can create an

imbalanced class distribution, making it harder for the model to learn the positive samples. Additionally, increasing the number of negatives does not contribute further useful information for training. This indicates that a model trained on a balanced version of a distantly supervised training dataset can achieve superior performance.

4.3 Threshold

Another important parameter when making predictions with the deep learning model is the probability acceptance threshold. When the dense layer of the model calculates a probability using the SoftMax function in the output, it creates a probability matrix indicating the probability that the input corresponds to each of the classes. The model will take the highest probability as the final class, which can be any of the HPO codes or the HP:None class, but with one condition, in case the class with the highest probability does not exceed a certain Threshold, then it will be assigned the HP:None class.

The correct choice of this threshold is something to be considered since a very low threshold could lead to many false positives, lowering the precision, while a very high and restrictive threshold could lead to an increase of false negatives, significantly reducing the recall. Figure 28 and Table 5 shows the results of several tests with different threshold values. The tests have been performed with the same parameters studied up to this section, using the combination of the dictionary system and the deep learning model.

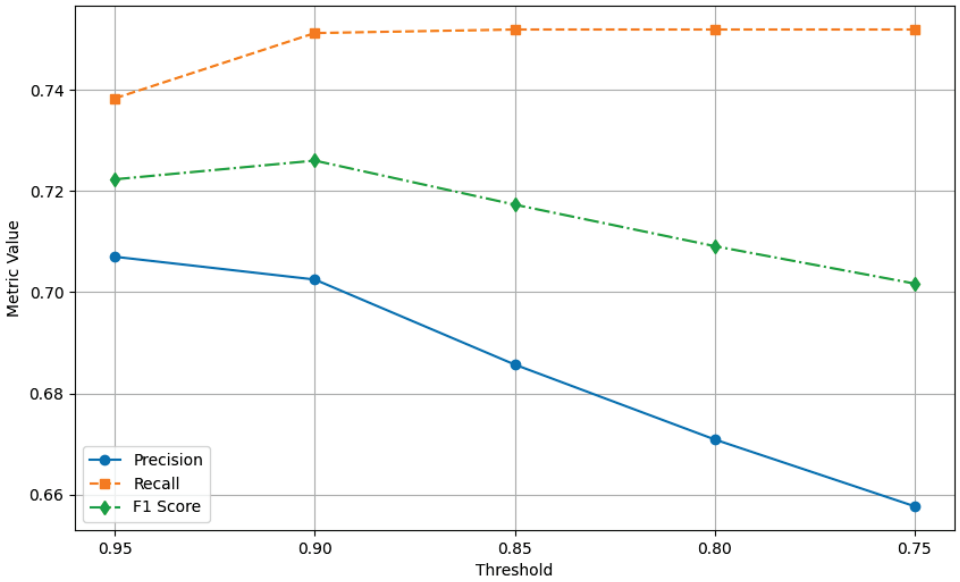

Figure 28. Metrics and threshold evolution

Threshold	Precision	Recall	F1 Score
0,95	**0.7070**	0.7383	0.7223
0,90	0.7025	0.7512	**0.7260**
0,85	0.6857	**0.7519**	0.7173
0,80	0.6709	**0.7519**	0.7091
0,75	0.6577	**0.7519**	0.7017

Table 5. Threshold testing results

In view of the results, we see that starting from a threshold of 0.95, as we reduce its value the precision is reduced and the recall increases, but as it is clearly seen it is at 0.90 where we obtain the best F1 score, giving up a little precision but gaining a lot of recall. In addition, there comes a time when, even if we lower the threshold further, the recall value does not increase, so we see that we only lose precision.

4.4 Lemmatization impact

On the dictionary side, it is interesting to study whether lemmatizing the dictionary generates a real impact or not. As mentioned in the methodology section, the dictionary is made up of the names, synonyms of the HPO concepts and additionally by the lemmatized versions of these instances.

It has been decided to test the construction of another dictionary consisting only of nouns and synonyms, without the lemmas, and to run the dictionary search in both dictionaries to check if there is a difference between the two approaches.

	Precision	Recall	F1 Score
Dictionary without lemmas	0.6375	0.5105	0.5670
Dictionary with lemmas	**0.6978**	**0.64499**	**0.6730**

Table 6. Dictionary lemmatization test results

As shown in Table 6, the difference between the two searches is very significant. The inclusion of lemmas in the dictionary improves both precision and recall in a very broad way, which at first glance might lead us to think that it would not have such an impact. The inclusion of greater variability of the dictionary phrases, together with the lemmatization of the input sentences, significantly increases the range of phenotypes found in the text, since with this, for example, we achieve that the variability of the plural and singular, or masculine and feminine does not affect recognition.

4.5 Data Augmentation impact

One of the latest techniques tested in the development process of this project has been data augmentation (explained in Section 3.9). Up to this point, all the models that have been shown and tested have been developed without the use of data augmentation for the study of parameters. The purpose of this section is to analyse the efficacy of the selected data augmentation technique, comparing it with the original models.

Similar to what happened with lemmatization, data augmentation allows us to further increase the number of HPO instances. Therefore, this new version of the HPO with augmented data has been used to build a new dictionary that has also undergone the process of lemmatization.

	Precision	Recall	F1 Score
Dictionary without DA	0.6978	0.6499	0.6730
Dictionary with DA	**0.7071**	**0.6941**	**0.7005**

Table 7. Data Augmentation applied to dictionary

Table 7 shows the result of doing dictionary lookup with the test texts and we see how increasing the number of instances with data augmentation causes a rise in both precision and recall. As in the previous section on lemmatization, a dictionary search will always benefit the more instances it has. However, since it is a technique that uses automatic translators that have not been checked manually, it could also introduce certain errors, although this is not reflected in the final metrics.

Similarly, with the new dictionary built with the augmented data, a new training dataset has been generated to generate a new model and compare it with the original model. his new model has been trained with the same parameters as the original model except for the number of negative instances used for training. For this case, as studied in Section 4.2, we have used 55.000 negative instances so that the dataset is balanced between negative and positive instances.

	Precision	Recall	F1 Score
Deep learning without DA	**0.7413**	**0.4324**	**0.5462**
Deep Learning with DA	0.6720	0.4276	0.5226

Table 8. Data Augmentation applied to the Deep Learning model

The results in Table 8 show that, unlike what happened in the dictionary search, the model trained also with the augmented data shows worse results than the original model. The Round-trip translation technique can introduce noise and errors into the data. If the translation and retranslation are not accurate, the resulting data may contain grammatical, semantic or contextual errors that were not present in the original data. This can confuse the model and cause it to learn incorrect patterns, thus decreasing its performance.

The last tests performed with data augmentation consisted of testing how well each of the dictionary and model versions worked together, both with and without data augmentation. We have already seen in other sections that these two approaches work best when they work in a complementary way, so several tests have been run to study the effectiveness of data augmentation from the perspective of the whole system.

	Dictionary DA			Dictionary without DA		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score
Deep Learning DA	0.6817	**0.7729**	0.7244	0.6812	0.7641	0.7203
Deep Learning without DA	0.7016	0.7655	**0.7321**	**0.7025**	0.7512	0.7260

Table 9. Hybrid system with data augmentation

The results in Table 9 show that the hybrid model that obtained the best recall was the one that used data augmentation in both the dictionary and deep learning. However, the model with the highest precision was the one that did not use data augmentation in any of the parts. This reaffirms that the data augmentation technique introduces certain errors that cause a decrease in the accuracy of the model, but at the same time it manages to recognize more concepts. The most balanced model in its metrics and with the highest F1 score is the one that uses data augmentation only for dictionary recognition but not for the deep learning model.

4.6 Deep Learning vs Dictionary

Finally, it is interesting to note once again that dictionary search and recognition with the deep learning model are very different. The limitation of dictionary models is that they are not able to recognize concepts that are not contained in the dictionary, and are sensitive to variations, synonyms and different ways of expressing concepts.

It is at this point that deep learning models try to provide this variability to complement the dictionary operation. What we can expect from the deep learning model in this specific problem is that it does not work particularly well

on its own due to the limited training data available, so the objective of this model is to work as an auxiliary model to the dictionary to complement the concept recognition.

	Precision	Recall	F1 Score
Dictionary with DA	0.7071	0.6941	0.7005
Deep Learning without DA	**0.7413**	0.4324	0.5462
Hybrid	0.7016	**0.7655**	**0.7321**

Table 10. Hybrid model results

Table 10 perfectly reflects the idea of this project, which is to achieve a dictionary search system that with the complementary help of a deep learning model can improve its performance. As we can see, we managed to improve the recall very significantly, recognizing variations not included in the dictionary, and with a precision very similar to that of the dictionary search.

The use of the dictionary search contributes significantly to improving completeness of the predictions due to its ability to recognize a wide variety of terms and concepts that have been previously defined. The dictionary acts as a solid knowledge base that can identify exact terms and their registered variations, which reduces omissions in the detection of relevant concepts. In addition, by combining the dictionary with the deep learning model, variations and synonyms that are not explicitly present in the dictionary can be detected. This combination allows the system to maintain high precision, as the dictionary provides an accurate reference, while the deep learning model improves the generalization capability and flexibility of the system to recognize concepts not previously defined achieving a balance between higher recall and good precision.

4.7 Final Model result

After the study carried out to find the correct configuration of the training and test parameters of the system, we can conclude that the most effective configuration of the tagging system is the following:

- Training epochs:110
- Number of negative instances: Balanced with the number of positive instances. In the case of the best model achieved is 35.000
- Batch size: 32
- Optimizer: Adam with 10^{-4} value of learning rate.
- Threshold: 0.90
- Dictionary created with Data Augmentation and lemmatization.

The hybrid system formed by the deep learning model and the dictionary search results in the following metrics when testing with the created test dataset:

	Precision	**Recall**	**F1 Score**
Hybrid	0.7016	0.7655	0.7321

Table 11. Final model results

Finally, a comparison of the results obtained by different state-of-the-art models discussed in section 2.7 for the GSC+ dataset has been performed.

	Precision	**Recall**	**F1 Score**
MetaMap	0.7070	0.5990	0.6490
PhenoBCBERT	0.7470	0.8130	0.7790
PhenoGPT	0.8090	0.8570	0.8320
PhenoBERT	0.8011	0.6698	0.7298
Phenotagger	0.7200	0.7600	0.7400
Spanish Tagger	0.7016	0.7655	0.7321

Table 12. Performance comparation of the State-of-the-Art models on GSC+ Dataset

The results obtained indicate that the model developed in this project achieves results that are on a par with the rest of the models, achieving an F1 score that is very similar and superior to some of the models. It should also be noted that the comparison is not entirely accurate since the other models have been tested on the original English version of the dataset while our model has been tested on the automatically translated version of the dataset.

The MetaMap is a dictionary-based tool and therefore the recall it obtains is much lower as it is not able to identify variations of phenotypes that are not in the dictionary. On the other hand, the other models differ basically in the language models used for the vector representations of the input phenotypes (BERT, BioBERT, Bio+Clinical BERT, Spanish BioBERT, GPT). The best-performing model is the PhenoGPT, which combines both a prompt-based learning and fine-tuning approach to improve recognition of phenotypes. This was to be expected as it is the most recent model that is state of the art right now and leverages the power of the GPT model.

5 Conclusions and Future Lines

5.1 Conclusions

The main objective of this work was to try to obtain a model that would achieve similar results for the context of phenotypes in Spanish to those that already existed in the state of the art for the case of phenotypes in English. This work has been based mainly on the PhenoTagger model because it was a simpler architecture to implement, which as a starting point for the Spanish case is a good approximation, and still achieving quite satisfactory results.

The initial objectives have been satisfactorily achieved experimentally:

1. An HPO dictionary has been constructed in Spanish that contains the labels, synonyms and generated lemmas of all the HPO concepts.

2. A deep learning model has been designed and trained based on a BERT model in Spanish chosen among several alternatives found in the state of the art that presents linguistic representations of the phenotypes of high quality.

3. A preprocessing system adapted to the Spanish case has been designed for the input texts.

4. It has been possible to design and study the quality of a hybrid model that has shown that the combination of dictionary search and the predictions of the deep learning model show better results together than separately.

5. It has been possible to implement a data augmentation technique that has significantly improved the results. This data augmentation using the round-trip translation technique has proven to work well for the dictionary search case but not for the deep learning model training due to the possible introduction of translation errors that confuse the model.

After an exhaustive analysis of both the state of the art and the methodology, and a subsequent testing phase, the most important conclusions of this work are as follows:

The lack of data: It has been one of the biggest challenges to overcome during development, both in training the deep learning model and in finding test data for the test phase. In any case, the resulting model has adapted well to the circumstances and has provided satisfactory results.

Improved recognition of phenotypic entities: The hybrid model developed, which combines dictionary search techniques and deep learning models, has been shown to significantly improve recall in the recognition of HPO entities. This is due to the ability of the deep learning model to recognize variations that are not included in the dictionary, while maintaining similar precision to dictionary search.

Impact of data augmentation: The technique of data augmentation using Round-Trip translation has been crucial in improving the performance of the model. This technique has allowed to increase the number of synonyms per HPO concept, resulting in a higher number of recognized entities, introducing some translation errors and noise that confound the deep learning model, but compensated by a higher recall of the dictionary search.

Balance between precision and recall: The results of the model show an adequate balance between precision and recall, with an F1 score of 0.7321. This balance is essential for practical applications in clinical context, where both precision and recall are crucial.

Optimal model configuration: Through extensive trial and error, it has been determined that the most effective configuration of the labeling system includes 110 training epochs, a batch size of 32, the use of the Adam optimizer with a learning rate of 10^{-4}, and a balanced train dataset with the same number of positive and negative instances. In addition, a threshold of 0.90 has been found to be the most optimal for the deep learning model.

5.2 Future Lines

Throughout the work, some limitations and possible improvements have been identified during the development of the project, which serve as future lines of work:

- **Expansion of the training corpus:** One of the current limitations is the limited amount of labeled data in Spanish. Increasing the training corpus with more labeled data could further improve the precision and recall of the model.

- **Test the use of fine tuning instead of feature generation**: Also retrain the BERT model with the training data, which requires more computational power, but is likely to make the model achieve more accurate results.

- **Use of other LLMs**: The use of other models with a larger number of parameters, and trained with larger corpora, can generate better representations that improve the model results.

- **Improve data augmentation techniques**: Continue to develop and refine data augmentation techniques to reduce errors introduced during the translation process and increase the variety of instances in the dictionary.

- **Integration with other healthcare systems**: Implement the model in real healthcare systems to evaluate its performance in a practical clinical setting and improve semantic interoperability between different EHR systems.

- **Multilingual model development**: Extend the model to work with clinical texts in multiple languages, thus improving its applicability in global healthcare applications.

- **Evaluation of other model architectures**: Explore other deep learning model architectures and NLP techniques that may offer additional improvements in phenotypic entity recognition.

6 Bibliography

[1] A. R. Aronson y F.-M. Lang, «An overview of MetaMap: historical perspective and recent advances», *J. Am. Med. Inform. Assoc.*, vol. 17, n.º 3, pp. 229-236, may 2010, doi: 10.1136/jamia.2009.002733.

[2] C. Jonquet, N. H. Shah, C. H. Youn, M. A. Musen, C. Callendar, y M.-A. Storey, «NCBO Annotator: Semantic Annotation of Biomedical Data», en *ISWC 2009 - 8th International Semantic Web Conference, Poster and Demo Session*, Washington, DC, United States, oct. 2009. Accedido: 22 de abril de 2024. [En línea]. Disponible en: https://hal.science/hal-04276274

[3] C. A. Deisseroth *et al.*, «ClinPhen extracts and prioritizes patient phenotypes directly from medical records to expedite genetic disease diagnosis», *Genet. Med.*, vol. 21, n.º 7, pp. 1585-1593, jul. 2019, doi: 10.1038/s41436-018-0381-1.

[4] C. Liu, F. S. Peres Kury, Z. Li, C. Ta, K. Wang, y C. Weng, «Doc2Hpo: a web application for efficient and accurate HPO concept curation», *Nucleic Acids Res.*, vol. 47, n.º W1, pp. W566-W570, jul. 2019, doi: 10.1093/nar/gkz386.

[5] A. P. Quimbaya *et al.*, «Named Entity Recognition Over Electronic Health Records Through a Combined Dictionary-based Approach», *Procedia Comput. Sci.*, vol. 100, pp. 55-61, ene. 2016, doi: 10.1016/j.procs.2016.09.123.

[6] «An approximate gazetteer for GATE based on levenshtein distance». Accedido: 15 de mayo de 2024. [En línea]. Disponible en: https://scholar.googleusercontent.com/scholar?q=cache:Sf7CnIMrQAQJ:scholar.google.com/+an+approximate+gazetter+for+gate+based+on+levenshtein+distance&hl=es&as_sdt=0,5

[7] Z. Huang, W. Xu, y K. Yu, «Bidirectional LSTM-CRF Models for Sequence Tagging». arXiv, 9 de agosto de 2015. doi: 10.48550/arXiv.1508.01991.

[8] R. Chalapathy, E. Z. Borzeshi, y M. Piccardi, «Bidirectional LSTM-CRF for Clinical Concept Extraction». arXiv, 25 de noviembre de 2016. doi: 10.48550/arXiv.1611.08373.

[9] J. Ji, B. Chen, y H. Jiang, «Fully-connected LSTM–CRF on medical concept extraction», *Int. J. Mach. Learn. Cybern.*, vol. 11, n.º 9, pp. 1971-1979, sep. 2020, doi: 10.1007/s13042-020-01087-6.

[10] «A Survey on Deep Learning for Named Entity Recognition | IEEE Journals & Magazine | IEEE Xplore». Accedido: 22 de abril de 2024. [En línea]. Disponible en: https://ieeexplore.ieee.org/abstract/document/9039685?casa_token=bAYqoii5ja8AAAAA:ayXT2tEp0xQCNcMfn4cnlzBgXXEL9qfF8dReT9_BJysdzhW3j1P0RCDLgKOs4OlJ_EsmpEGzog

[11] «Building a Named Entity Recognition model using a BiLSTM-CRF network». Accedido: 22 de abril de 2024. [En línea]. Disponible en: https://domino.ai/blog/named-entity-recognition-ner-challenges-and-model

[12] «Text Chunking Using Transformation-Based Learning | SpringerLink». Accedido: 22 de abril de 2024. [En línea]. Disponible en: https://link.springer.com/chapter/10.1007/978-94-017-2390-9_10

[13] Y. Lou, X. Zhu, y K. Tan, «Dictionary-based matching graph network for biomedical named entity recognition», *Sci. Rep.*, vol. 13, n.º 1, p. 21667, dic. 2023, doi: 10.1038/s41598-023-48564-w.

[14] B. Min *et al.*, «Recent Advances in Natural Language Processing via Large Pre-trained Language Models: A Survey», *ACM Comput. Surv.*, vol. 56, n.º 2, p. 30:1-30:40, sep. 2023, doi: 10.1145/3605943.

[15] H. Wang, J. Li, H. Wu, E. Hovy, y Y. Sun, «Pre-Trained Language Models and Their Applications», *Engineering*, vol. 25, pp. 51-65, jun. 2023, doi: 10.1016/j.eng.2022.04.024.
[16] J. Devlin, M.-W. Chang, K. Lee, y K. Toutanova, «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». arXiv, 24 de mayo de 2019. doi: 10.48550/arXiv.1810.04805.
[17] J. Pennington, R. Socher, y C. Manning, «GloVe: Global Vectors for Word Representation», en *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, y W. Daelemans, Eds., Doha, Qatar: Association for Computational Linguistics, oct. 2014, pp. 1532-1543. doi: 10.3115/v1/D14-1162.
[18] D. Jatnika, M. A. Bijaksana, y A. A. Suryani, «Word2Vec Model Analysis for Semantic Similarities in English Words», *Procedia Comput. Sci.*, vol. 157, pp. 160-167, ene. 2019, doi: 10.1016/j.procs.2019.08.153.
[19] P. Rajpurkar, J. Zhang, K. Lopyrev, y P. Liang, «SQuAD: 100,000+ Questions for Machine Comprehension of Text». arXiv, 10 de octubre de 2016. doi: 10.48550/arXiv.1606.05250.
[20] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, y S. R. Bowman, «GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding». arXiv, 22 de febrero de 2019. doi: 10.48550/arXiv.1804.07461.
[21] J. Lee *et al.*, «BioBERT: a pre-trained biomedical language representation model for biomedical text mining», *Bioinformatics*, vol. 36, n.º 4, pp. 1234-1240, feb. 2020, doi: 10.1093/bioinformatics/btz682.
[22] «🎈 The Evolution of Transformers and LLM (Large Language Models): 💡Roadmap, Impact, and Future Prospects | Kaggle». Accedido: 12 de junio de 2024. [En línea]. Disponible en: https://www.kaggle.com/discussions/general/a
[23] «Hello GPT-4o». Accedido: 11 de junio de 2024. [En línea]. Disponible en: https://openai.com/index/hello-gpt-4o/
[24] «ChatGPT». Accedido: 11 de junio de 2024. [En línea]. Disponible en: https://chatgpt.com
[25] «Meta Llama 3», Meta Llama. Accedido: 11 de junio de 2024. [En línea]. Disponible en: https://llama.meta.com/llama3/
[26] «Gemini - Chatea para dar rienda suelta a tus ideas», Gemini. Accedido: 20 de junio de 2024. [En línea]. Disponible en: https://gemini.google.com
[27] M. Bayer, M.-A. Kaufhold, y C. Reuter, «A Survey on Data Augmentation for Text Classification», *ACM Comput. Surv.*, vol. 55, n.º 7, p. 146:1-146:39, dic. 2022, doi: 10.1145/3544558.
[28] «Human Phenotype Ontology in 2021 | Nucleic Acids Research | Oxford Academic». Accedido: 22 de abril de 2024. [En línea]. Disponible en: https://academic.oup.com/nar/article/49/D1/D1207/6017351?login=false
[29] «Human Phenotype Ontology». Accedido: 13 de mayo de 2024. [En línea]. Disponible en: https://hpo.jax.org/app/about
[30] «Example of hierarchical (tree) structure of data in the Human Phenotype... | Download Scientific Diagram». Accedido: 14 de mayo de 2024. [En línea]. Disponible en: https://www.researchgate.net/figure/Example-of-hierarchical-tree-structure-of-data-in-the-Human-Phenotype-Ontology-HPO_fig1_337324826
[31] L. Luo *et al.*, «PhenoTagger: a hybrid method for phenotype concept recognition using human phenotype ontology», *Bioinformatics*, vol. 37, n.º 13, pp. 1884-1890, jul. 2021, doi: 10.1093/bioinformatics/btab019.

[32] Y. Feng, L. Qi, y W. Tian, «PhenoBERT: A Combined Deep Learning Method for Automated Recognition of Human Phenotype Ontology», *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 20, n.º 2, pp. 1269-1277, 2023, doi: 10.1109/TCBB.2022.3170301.
[33] J. Yang *et al.*, «Enhancing phenotype recognition in clinical notes using large language models: PhenoBCBERT and PhenoGPT», *Patterns N. Y. N*, vol. 5, n.º 1, p. 100887, ene. 2024, doi: 10.1016/j.patter.2023.100887.
[34] «Welcome to Python.org», Python.org. Accedido: 24 de mayo de 2024. [En línea]. Disponible en: https://www.python.org/
[35] «TensorFlow», TensorFlow. Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://www.tensorflow.org/?hl=es-419
[36] «🤗 Transformers». Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://huggingface.co/docs/transformers/es/index
[37] «NumPy -». Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://numpy.org/
[38] «Matplotlib — Visualization with Python». Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://matplotlib.org/
[39] «NLTK :: Natural Language Toolkit». Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://www.nltk.org/
[40] «spaCy · Industrial-strength Natural Language Processing in Python». Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://spacy.io/
[41] «Translate text | English | DeepL API Docs». Accedido: 1 de junio de 2024. [En línea]. Disponible en: https://developers.deepl.com/docs/api-reference/translate
[42] «Anaconda | The Operating System for AI», Anaconda. Accedido: 27 de mayo de 2024. [En línea]. Disponible en: https://www.anaconda.com/
[43] «Spanish · spaCy Models Documentation», Spanish. Accedido: 29 de mayo de 2024. [En línea]. Disponible en: https://spacy.io/models/es
[44] E. Fredkin, «Trie memory», *Commun. ACM*, vol. 3, n.º 9, pp. 490-499, sep. 1960, doi: 10.1145/367390.367400.
[45] «Trie Data Structure». Accedido: 17 de junio de 2024. [En línea]. Disponible en: https://www.thealgorist.com/Algo/Trie
[46] «Home», PubMed Central (PMC). Accedido: 29 de mayo de 2024. [En línea]. Disponible en: https://www.ncbi.nlm.nih.gov/pmc/
[47] D. C. Comeau, C.-H. Wei, R. Islamaj Doğan, y Z. Lu, «PMC text mining subset in BioC: about three million full-text articles and growing», *Bioinformatics*, vol. 35, n.º 18, pp. 3533-3535, sep. 2019, doi: 10.1093/bioinformatics/btz070.
[48] E. Sayers, «A General Introduction to the E-utilities», en *Entrez Programming Utilities Help [Internet]*, National Center for Biotechnology Information (US), 2022. Accedido: 29 de mayo de 2024. [En línea]. Disponible en: https://www.ncbi.nlm.nih.gov/books/NBK25497/
[49] «Helsinki-NLP/opus-mt-es-en · Hugging Face». Accedido: 29 de mayo de 2024. [En línea]. Disponible en: https://huggingface.co/Helsinki-NLP/opus-mt-es-en
[50] «Helsinki-NLP/opus-mt-en-es · Hugging Face». Accedido: 29 de mayo de 2024. [En línea]. Disponible en: https://huggingface.co/Helsinki-NLP/opus-mt-en-es
[51] T. Groza *et al.*, «Automatic concept recognition using the Human Phenotype Ontology reference and test suite corpora», *Database*, vol. 2015, p. bav005, ene. 2015, doi: 10.1093/database/bav005.
[52] L. M, L. A, y C. Fm, «Identifying Human Phenotype Terms by Combining Machine Learning and Validation Rules», *BioMed Res. Int.*, vol. 2017, 2017, doi: 10.1155/2017/8565739.