PLN_PPM_ISB at MentalRiskES 2024: Detection of Gambling Disorders and Type of Addiction

Pedro Pablo Molina[1,*], Isabel Segura-Bedmar[1]

[1]*Human Language and Accessibility Technologies Group (HULAT), Computer Science and Engineering Department, University Carlos III of Madrid (UC3M), Leganés*

Abstract
The following paper describes the solution to the two tasks presented by MentalRiskES 2025 [1], a competition dedicated to the early detection of mental disorders. In this case, the tasks were focused on detecting problems related to gambling addiction. During the design of the solutions, we aimed to explore various approaches to better adapt to the environment in which the data was collected. For the first task, we used: SVM, SVM with zero-shot assistance, and a transformer model. For the second task, we applied Data Augmentation for an SVM model and for two transformer models. However, the primary objective of this paper is to demonstrate the different techniques used, their limitations, and the conclusions drawn from the experiments.

Keywords
Early Risk Detection, Support Vector Machine, Transformer, Data Augmentation

1. Introduction

Mental illnesses play a very important role in people's lives. As reflected in the description of MentalRiskES, after COVID-19, there was a significant increase in cases of anxiety and severe depression. Furthermore, the World Health Organization (WHO) states that these conditions increased by 26% and 28% respectively in just one year [2].

Gambling addiction, or gambling disorder, is one of the addictions that has the greatest impact on our society, especially due to the ease of access among young people. Data from the Ministry of Consumer Affairs reveals that 12.45% of young people aged 18 to 25 who have participated in online betting develop gambling problems [3]. In other words, the risks increase considerably when it comes to an online environment. This may be due to the ease and speed with which it can be done, as well as the lack of proper money valuation, since it is only perceived as a number on the screen.

Therefore, early detection of this type of disorder is very important. The use of Natural Language Processing (NLP) is a promising tool for identifying signs of such disorders. Initiatives like eRisk 2021[4], organized by the Cross-Language Evaluation Forum (CLEF), and MentalRiskES [1], within the framework of IberLEF [5], promote the development of systems capable of detecting gambling addiction risks.

In the 2025 edition of MentalRiskES, the competition consisted of two tasks [1]: (1) Risk Detection of Gambling Disorders and (2) Detection of the Type of Addiction. These tasks used texts collected from platforms like Telegram or Twitch, where users spontaneously communicate their emotions [6].

For task 1, we considered different approaches:

1. For our initial approach, we represented texts using TF-IDF. We then applied classic algorithms, such as Support Vector Machines (SVM) [7], as they usually perform well in text classification tasks [8].
2. The second approach augmented this baseline SVM and TF-IDF method by incorporating an additional feature. This feature was derived using a zero-shot model, specifically vicgalle/xlm-

IberLEF 2025, September 2025, Zaragoza, Spain
*Corresponding author.
pedropmolinaa@gmail.com (P. P. Molina); isegura@inf.uc3m.es (I. Segura-Bedmar)
0009-0002-3131-8591 (P. P. Molina); 0000-0001-7116-9338 (I. Segura-Bedmar)

roberta-large-xnli-anli [9]. The model was applied with the labels ”sano” (healthy) and ”enfermo” (ill) to classify user messages from a health user and those from a patient with a mental condition.
3. Finally, we used the pre-trained model myahan007/bert-base-spanish-wwm-cased-finetuned-tweets, model based on BERT [10] that was pretrained using a Whole Word Masking strategy on a Spanish corpus derived from tweets and other informal texts [11], in which different training data were considered using Data Augmentation.

For task 2:

1. An SVM model with Data Augmentation was used for the lootboxes class since it contained very few data samples.
2. We used the same training dataset as the previous task but used the pre-trained model myahan007/bert-base-spanish-wwm-cased-finetuned-tweets [10].
3. Finally, the pre-trained model myahan007/bert-base-spanish-wwm-cased-finetuned-tweets [10] was used, but Data Augmentation was applied to several classes, better balancing them, considering that generalization could lead to better results when facing new messages.

Those are mainly the runs we submitted, but we will later discuss in more detail all the attempts we made to achieve them.

2. Description of Dataset and Tasks

The competition provided a series of messages sent by different users, both on Telegram and Twitch. As mentioned earlier, these platforms are designed for the constant exchange of information, with minimal filters, and their importance has grown significantly in recent years.

A single dataset was provided for both tasks, containing all messages from each user along with their label, applicable to both Task 1 and Task 2.

The purpose of this competition is to identify mental disorders at an early stage. To simulate a realistic environment, the organization implemented a server that emulates an authentic conversation, delivering data packets containing one message per user. The system must assign a label to each user based on both the current message and all previous messages before receiving the next packet. The ultimate goal is to predict the possible mental disorder of each user as quickly as possible.

2.1. Task 1: Risk Detection of Gambling Disorders

This is a binary classification task aimed at determining whether a user is at high risk (label = 1) or low risk (label = 0) of developing a gambling-related disorder based on their messages. The objective is to enable early detection and facilitate timely interventions. Table 1 shows the class distribution for Task 1 for the trial and training datasets provided by the organizers.

Table 1
Class distribution for task 1

	High Risk		**Low Risk**	
	Subjs.	**Msgs.**	**Subjs.**	**Msgs.**
Trial	3	229	4	215
Train	172	11052	178	11439
Total	175	11281	182	11654

2.2. Task 2: Type of Addiction Detection

This is a multi-label classification task aimed at determining the specific type of addiction associated with the disorder. The available labels are Betting, Online Gaming, Trading, and Lootboxes. The class distribution is shown in Table 2.

Table 2
Class distribution for Task 2

	Betting		Online Gaming		Trading		Lootboxes	
	Subjs.	**Msgs.**	**Subjs.**	**Msgs.**	**Subjs.**	**Msgs.**	**Subjs.**	**Msgs.**
Trial	2	260	2	25	2	151	1	8
Train	85	9566	104	2703	135	10007	26	215
Total	87	9826	106	2728	137	10158	27	223

3. System architecture and Techniques

Due to the characteristics of the dataset, it was decided to train all models with concatenated messages, separated by a line break. Additionally, it is important to consider that this approach may result in very long message lengths, so we tested different models capable of handling varying input sizes.

3.1. Data Augmentation

Due to the small size of the dataset, it was decided to use the Data Augmentation (DA) technique, which involves the artificial creation of training data for machine learning through transformations. This is a widely studied research field across various machine learning disciplines [12]. In the case of Natural Language Processing (NLP), there are various techniques available. In this context, we will differentiate between Task 1 and Task 2, as the classes in each task are different. Our goal is to generate specific types of data tailored to each task's unique requirements.

3.2. Task 1: Risk Detection of Gambling Disorders

In this case, we will apply three techniques:

- **Back Translation**: is a common DA technique that translates a sentence into another language and then back to the original language [13]. This helps generate paraphrased versions of the original texts without changing their semantic meaning. We used the MarianMT models from the Helsinki-NLP project, specifically Helsinki-NLP/opus-mt-es-en [14] and Helsinki-NLP/opus-mt-en-es [15] for Spanish-English-Spanish translation. This method is particularly effective in improving generalization for text classification tasks by increasing data diversity without manual annotation [16].
- **Summarization**: summarization-based augmentation involves using a language model to compress messages while preserving their most relevant information. In this context, we employed prompting using LLMs such as filipealmeida/Mistral-7B-Instruct-v0.1-sharded [17] and NousResearch/Nous-Hermes-2-Mistral-7B-DPO [18]. Each input text was passed with an instruction to summarize the message while retaining the semantic intent associated with its labeled class. This technique serves to create concise variants of original messages, which helps the model generalize by focusing on the core content [19].
- **LLM prompting**: Prompts were specifically designed to simulate user messages from informal platforms (e.g., Twitch or Discord) related to high and low risk. We used the model NousResearch/Nous-Hermes-2-Mistral-7B-DPO, which is optimized using Direct Preference Optimization (DPO) and trained on high-quality synthetic datasets. This method provides high-variability, realistic texts aligned with the characteristics of each class [20].

These data augmentation techniques are widely recognized as effective strategies in scenarios with limited training data, as they help increase dataset diversity and improve model generalization [21]. However, in our experiments, we observed that these methods did not significantly improve our models' performance. This is mainly due to the subtle semantic overlap between texts labeled as high and low risk, which makes it challenging for augmented data to enhance class discrimination. Notably, the most effective technique in our case was LLM prompting. For instance, back translation sometimes altered or lost the meaning of key words critical for accurate class identification, reducing its effectiveness in distinguishing between classes.

3.3. Task 2: Type of Addiction Detection

In this case, considering the classes we need to predict and the imbalance between classes, the best approach is to generate synthetic data using the previously mentioned LLM prompting technique.

We used the Hugging Face model NousResearch/Nous-Hermes-2-Mistral-7B-DPO, which is a flagship 7B model improved through DPO (Direct Preference Optimization). This model has shown significant improvements across multiple benchmarks, including AGIEval, BigBench Reasoning, GPT4All, and TruthfulQA. It was previously trained on 1,000,000 instructions and chats of GPT-4 quality or higher, primarily using synthetic data and other high-quality datasets. This choice was made due to its ability to generate realistic and contextually appropriate texts for the task.

First, we did it for the lootboxes class with the following parameters:

- Temperature (0.95), Top-k (60), Top-p (0.95), Repetition Penalty (1.1): Configured to ensure diversity and realism.
- Minimal cleaning: Removal of dashes and quotation marks at the beginning.

This was the prompt translate from Spanish to English:

prompt_base = ””” Generate a message from a user in a chat while playing video games with lootboxes. Do not literally mention “boxes” all the time. Use phrases that express frustration, excitement, or addiction when opening random rewards. The style should be very informal, like Twitch or Discord: use expressions like ”xd”, ”haha”, ”I blew all my money”, ”one more”, ”this is a scam”, etc. You can mention: knives, skins, stickers, drops, boosters, packs, legendary drops, credits, gems, stattrak, foil items, trash or epic rewards, duplicates, odds, pity counter, upgrades, reroll, or any typical lootbox dynamics. It can also reference games like CS:GO, FIFA, Overwatch, Clash Royale, Valorant, Fortnite, etc. Message: ”””

This prompt simulates typical messages from players on platforms like Twitch or Discord while interacting with lootboxes. The style is informal and expressive, including phrases like ”xd”, ”haha”, ”this is a scam”, and references to common items: skins, knives, trash rewards, duplicates, probabilities. It also mentions games like CS:GO, FIFA, Overwatch, and Valorant.

Similarly, this process was carried out for online gaming and betting, with the same intention but adjusted to the characteristics of each class:

prompt_base = ””” Generate a message from a user in a chat while participating in online gambling (online gaming). Do not use the word “casino” in every message. It should show frustration or addiction with a very informal style, as if chatting on Twitch or Discord: ”xd”, ”the slot machine doesn’t pay anything”, ”it gave me 50 free spins and I got 10 cents”, ”my entire salary went on blackjack”, ”one more round and I’m done (or not haha)”. You can mention: roulette, slot machines, blackjack, jackpots, welcome bonuses, free spins, multipliers, mega wins, wilds, dead spins, etc. Message: ”””

prompt_base = ””” Generate a message from a user in a chat while participating in sports betting. Do not repeat the word “bet” all the time. The message should reflect emotions like frustration, euphoria, or addiction when betting money on sports events, especially football. The style should be very informal, as if speaking on Twitch or Discord: use expressions like ”xd”, ”I blew it on the combo”, ”VAR ruined me again”, ”Haaland saved me”, ”haha this parlay is hopeless”. You can mention odds, combos, penalties, live betting, VAR, goals in the 90+3, corner bets, cards, cashout, losing by one goal or winning by a miracle, etc. Message: ”””

These prompts were designed to generate realistic, context-specific messages, preserving the unique linguistic features of each class while maintaining an informal and spontaneous conversational style.

3.4. Approaches for Task 1: Risk Detection of Gambling Disorders

During the development phase, to properly train and compare our model, we used the same dataset, dividing the combined trial+train set into training (80%) and test (20%) subsets.

Table 3
Distribution train_test partition task 1

	train	**test**
high risk	140	35
low risk	145	37
Total	285	72

In this case, stratify was not used for class balancing since the dataset is already quite well balanced.

3.4.1. Classical Machine Learning Classifiers

Classical algorithms represent an effective alternative for text classification, which is why it was decided to give them a chance. Additionally, they are capable of adapting to any text length.

Firstly, it was decided to experiment with different classical models using **TF-IDF** and a basic preprocessing technique, in which the text was converted to lowercase, stopwords and non-alphabetic characters were removed. The SnowballStemmer algorithm was applied to obtain the root of each word.

Table 4
Results for Task 1 using classical classifiers without hyperparameters. The scores are obtained on our test split.

	accuracy	**precision**	**recall**	**f1-score**
SVM	**0.666667**	**0.668750**	**0.666667**	**0.665635**
KNN	0.638889	**0.685185**	0.638889	0.614815
Decision Tree	0.574074	0.575736	0.574074	0.571724
Random Forest	0.592593	0.601504	0.592593	0.583450
Gradient Boost	0.620370	0.621411	0.620370	0.619555

We can see that the SVM model already stands out. We also decided to perform hyperparameter tuning with RandomSearch using F1-score as the objective metric to truly confirm if it is the best, and we obtained:

Table 5
Results for Task 1 using classical classifiers with hyperparameters task 1. The scores are obtained on our test split.

	accuracy	**precision**	**recall**	**f1-score**
SVM	**0.703704**	**0.703984**	**0.703704**	**0.703602**
KNN	0.638889	0.685185	0.638889	0.614815
Decision Tree	0.574074	0.575736	0.574074	0.571724
Random Forest	0.592593	0.601504	0.592593	0.583450
Gradient Boost	0.620370	0.621411	0.620370	0.619555

Despite performing hyperparameter tuning, after analyzing the results, we decided to further explore the SVM model to achieve the best classifier by using different combinations of preprocessing and hyperparameters.

Different text preprocessing strategies were tested and combined. Below we describe the selected approaches along with the motivation behind each of them.

- **Lemmatization with SpaCy:** This approach aims to retain the linguistically accurate form of each word, which is especially useful in tasks where lexical meaning is important. The SpaCy library was used for this purpose [22], as it supports lemmatization while considering both grammatical category and syntactic context. Standard normalization steps were applied beforehand: lowercasing, tokenization via `nltk.word_tokenize()` [23], stopword removal (using `nltk.corpus.stopwords`), and filtering of non-alphabetic tokens. This approach was particularly useful for evaluating models based on richer semantic representations, such as TF-IDF [24].

- **Stemming with SnowballStemmer:** The SnowballStemmer algorithm from NLTK was used [23], configured for Spanish. Although stemming does not guarantee valid words in the language, it can be advantageous for frequency-focused methods such as bag-of-words by reducing vocabulary variability [25]. The same preprocessing steps as in the previous method were applied, replacing lemmatization with stemming to allow a direct comparison in terms of performance and processing speed.
- **Stemming with TweetTokenizer:** Given that the competition dataset includes brief and informal messages, often with social media expressions, a variant using NLTK's TweetTokenizer [23] was tested. This tokenizer is designed to preserve symbols like hashtags, mentions, contractions, and emojis. It was combined with the same stemming algorithm (SnowballStemmer) and stopword removal, aiming for a preprocessing setup better adapted to short and colloquial texts [26].
- **Emoji processing without alphanumeric cleaning:** To assess the role emojis may play in risk detection, a configuration was tested where no special character removal or morphological transformations were applied. Instead, the `emoji` library was used to convert each emoji into its textual description in Spanish using `emoji.demojize()` [27]. This approach was intended to evaluate the specific contribution of emojis to model performance.
- **Stopword removal only with TweetTokenizer:** Finally, a minimal configuration was considered in which only Spanish stopwords were removed after applying TweetTokenizer. This setting was included as a simple baseline to compare against more complex techniques such as lemmatization or emoji handling, in order to assess their actual contribution to system performance.

These five configurations were selected after testing various combinations and were deemed sufficiently representative of diverse text preprocessing strategies—from linguistically oriented approaches to ones adapted to informal social media language. They also offered a good trade-off between simplicity, execution time, and the ability to capture relevant information in the context of the MentalRiskES 2025 competition.

To find the best hyperparameters, we used GridSearch with a relatively small search space.

- **C**: [0.1, 1, 10]
- **kernerl**: ['linear', 'rbf']
- **gamma**: ['scale', 'auto']
- **tol**: [1e-3, 1e-2, 1e-1]

After combining both, we obtained the following results:

Table 6
Results for Task 1 using SVM with preprocess and hyperparameters on our test split

Preprocess	Accuracy	F1-Score	Best Hyperparameters
lemmatization	0.680556	0.684932	`C: 10, gamma: scale, kernel: rbf, tol: 0.001`
stemming	0.666667	0.657143	`C: 10, gamma: scale, kernel: rbf, tol: 0.1`
tweet_stemming	0.694444	0.685714	`C: 10, gamma: scale, kernel: rbf, tol: 0.001`
emoji	**0.708333**	**0.704225**	`C: 10, gamma: scale, kernel: rbf, tol: 0.001`
stopwords_only	**0.708333**	**0.704225**	`C: 10, gamma: scale, kernel: rbf, tol: 0.001`

We will stick with stopwords_only since the processing time and computational cost are lower than those of emoji.

3.4.2. Using zero-shot Classifier

After attempting Data Augmentation without improving the results, we decided to analyze the options that seemed most promising.

In this context, the idea of applying a pipeline with a transformer for text classification using zero-shot emerged. The goal was to classify each user's messages one by one, where 0 indicates that the message does not suggest gambling addiction, and 1 indicates that it does. This approach was conceived to enable early detection of the moment when a high risk of gambling addiction begins to appear.

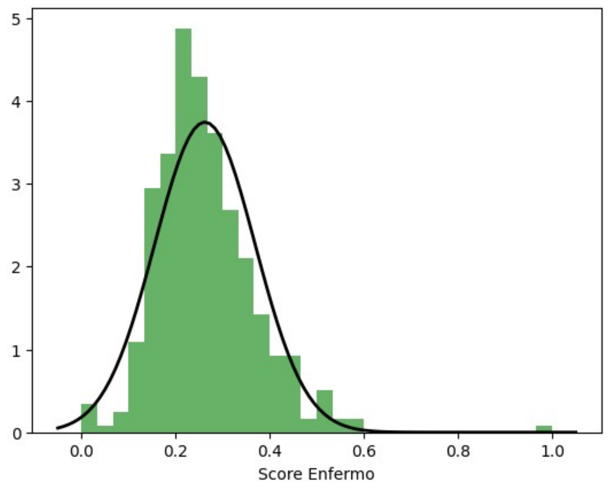
After extensive research, we did not find any suitable model specifically dedicated to gambling addiction detection. Therefore, we decided to use the vicgalle/xlm-roberta-large-xnli-anli model from Hugging Face [28], with the labels ”sano” and ”enfermo”.

Finally, for each user, we stored all messages together and decided to add the score_enfermo column, where we calculated the ratio of messages labeled as ”sick” to the total number of messages for that user.

With this setup, we conducted a study to visualize whether there is any correlation between risk and score_enfermo, thereby validating our creative approach.

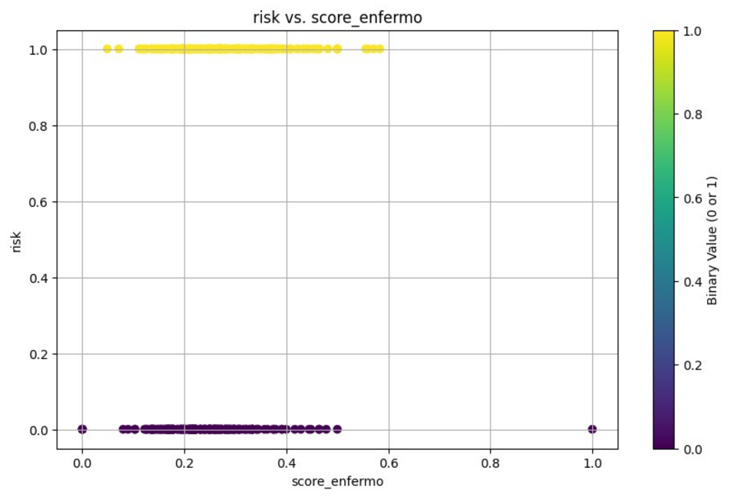
- **Point-biserial correlation coefficient**: It is a special case of the Pearson correlation coefficient used to measure the relationship between a binary variable and a continuous variable. It is equivalent to calculating the Pearson correlation coefficient directly between the binary variable (encoded as 0 and 1) and the continuous variable. Intuitively, it measures how much the mean of the continuous variable differs between the two groups defined by the binary variable.

  Figure 1: Fit to the normal distribution of the score_enfermo distribution

  The point-biserial correlation coefficient of 0.19 indicates a weak positive correlation between the binary list (0 and 1) and the list of probabilities. This means that there is a slight tendency for the continuous variable (probabilities) to have slightly higher values when the binary variable is 1 compared to when it is 0. However, this relationship is not very strong. The p-value of 0.0003 is very low (generally considered significant if it is less than 0.05). This indicates that the correlation you observed (although weak) is statistically significant. In other words, it is unlikely that you would have obtained a correlation of 0.19 by chance if there were no real (albeit small) relationship between the two lists in the population from which your data originate.
- **Degree of dispersion**: Another way to check for a relationship is by using a graph that shows the distribution of scores as a function of the risk field. In this case, the graph does not provide much insight.
- **ROC Curve**: It allows us to determine the optimal threshold. In this case, using roc_curve from sklearn, we find that the optimal threshold is 0.299. This means that if the value of score_enfermo is greater than 0.299, there appears to be a relationship indicating that the user is indeed at risk.

Figure 2: Degree of dispersion risk vs score_enfermo



After this, we used the score_enfermo column in the SVM. We performed the vectorization of the messages using TF-IDF to obtain a matrix with textual features. Then, we concatenated the score_enfermo column to this matrix using pd.concat(), thus creating a feature set where each row includes both the textual information and the score_enfermo value.

Table 7
Results from SVM with preprocess, score_enfermo and hyperparameters task 1

Preprocess	Accuracy	F1-Score	Best Hyperparameters
lemmatization	0.680556	0.684932	`C: 10, gamma: scale, kernel: linear, tol: 0.01`
stemming	0.722222	0.722222	`C: 10, gamma: scale, kernel: rbf, tol: 0.001`
tweet_stemming	0.736111	0.732394	`C: 10, gamma: scale, kernel: rbf, tol: 0.1`
emoji	0.708333	0.704225	`C: 10, gamma: scale, kernel: rbf, tol: 0.1`
stopwords_only	**0.75**	**0.75**	`C: 10, gamma: scale, kernel: rbf, tol: 0.01`

As we can see, the results have improved in almost all preprocessing methods.

3.4.3. Transformers Models

Finally, we chose to use Transformer models in this task due to their ability to capture the complex context and linguistic nuances present in the messages, especially in Spanish texts from social media.

Given the size, language, and colloquial expressions typical of these texts, we decided to experiment with the following Transformer models:

- **bert-base-uncased** [29]: This is the original version of the BERT model (*Bidirectional Encoder Representations from Transformers*) [30], trained on BookCorpus and English Wikipedia. It has 12 transformer layers, 12 attention heads, and a hidden size of 768 per token, totaling 110 million parameters. It is an uncased model, meaning it does not distinguish between uppercase and lowercase letters. Although it is not adapted to Spanish, it was included as a baseline due to its widespread use and proven effectiveness in many English text classification tasks. Its inclusion provides a general reference point for comparison with more specialized models.
- **myahan007/bert-base-spanish-wwm-cased-finetuned-tweets** [31]: This model is based on *dccuchile/bert-base-spanish-wwm-cased*, a Spanish adaptation of BERT [30] trained with Whole Word Masking (WWM) on Spanish Wikipedia and other corpora. The version used here was further fine-tuned on a corpus of tweets, making it especially suitable for this task, since the dataset consists of informal, abbreviated, and colloquial messages from social platforms. Its use is justified by its better adaptation to real-world, non-standard language.

- **bertin-project/bertin-roberta-base-spanish** [32, 33]: This model was trained from scratch using the RoBERTa architecture (*Robustly Optimized BERT Pretraining Approach*) [34] on the mc4-es corpus, which contains large volumes of Spanish text from Common Crawl. Unlike BERT, RoBERTa removes tasks like Next Sentence Prediction and employs longer, dynamic training. This monolingual Spanish variant allows us to assess the benefits of specific pretraining in Spanish without multilingual interference and under a more robust architecture. Its inclusion helps evaluate whether training from scratch in Spanish provides an advantage over multilingual or adapted models.
- **PlanTL-GOB-ES/RoBERTa-base-bne** [35]: Developed by Spain's National Language Technologies Plan (PlanTL), this model also uses the RoBERTa architecture [34], but was trained on CORPES XXI and other high-quality linguistic resources from the National Library of Spain. It aims to faithfully represent contemporary normative Spanish, so it was included to compare its performance against models adapted to more informal registers. Its linguistic quality and carefully curated training data make it a solid reference for tasks in standard Spanish.
- **distilbert-base-multilingual-cased** [36]: DistilBERT [37] is a compressed version of BERT [30] created via knowledge distillation, reducing the number of layers to 6 while maintaining 12 attention heads and a hidden size of 768, resulting in a 40% smaller model. The multilingual version used was trained on over 100 languages. Although not specialized in Spanish or social texts, its smaller size allows for greater computational efficiency, making it a practical alternative in resource-constrained environments. Its inclusion allows us to assess the trade-off between performance and efficiency in multilingual tasks.

Together, the selected models span different approaches: general vs. specialized, monolingual vs. multilingual, and training on normative vs. informal texts.

Experiments using transformer-based models relied on default hyperparameters, though fine-tuning was applied as described in Table **??**. This process involves adding an output layer (e.g., softmax for binary or multi-class classification) and training the model on the competition-provided message dataset. This technique enables strong results even on small datasets by leveraging the model's pretraining knowledge.

Due to the dataset's limited size, the same fine-tuning setup was used across all models to avoid overfitting and ensure fair comparison. Additionally, a `TrainerCallback` was used to manage early stopping.

Table 8
Training hyperparameters for transformers in task 1

Hyperparameters	**Value**
Learning Rate	2e-5
Batch Size	16
Seed	100
Max Length	512
Epochs	3
Patience	5
Evaluation Strategy	Steps
Eval Steps	100
Metric for Best Model	eval_loss
Scheduler Type	Linear
Warmup Ratio	0.1

3.4.4. Transformer Models

Table 9 presents the results obtained by each transformer model.

Table 9
Performance of Transformer models in Task 1 (accuracy, precision, recall, F1-score)

Model	Accuracy	F1-Score	Precision	Recall
bert-base-uncased	0.59722	0.57344	**0.65508**	0.61223
myahan007/bert-base-spanish-wwm-cased-finetuned-tweets	**0.65278**	**0.65367**	0.65402	**0.65271**
bertin-project/bertin-roberta-base-spanish	0.48611	0.43833	0.50652	0.50387
PlanTL-GOB-ES/RoBERTa-base-bne	0.47222	0.32075	0.23611	0.5
distilbert-base-multilingual-cased	0.59722	0.59338	0.61111	0.60449

In this case, the results were not as good as expected. Models better adapted to informal language, such as the tweet-based Spanish model, clearly outperformed others pretrained on normative corpora or in different languages. This supports the idea that domain and text-type adaptation is crucial [38].

Additionally, generalist models or those trained on cleaner corpora tend to underperform when dealing with noisy texts, emojis, or abbreviations. This suggests that specialized pretraining can yield significant gains without complex adjustments. These models are likely better at understanding tone, intent, and idiosyncrasies in social media messages—important indicators of risk in this task.

Multilingual or compact models like *distilbert-base-multilingual-cased* showed decent performance, which is promising for low-resource settings. However, they still lag behind the tweet-specialized Spanish model.

Finally, note that transformer models are limited to sequences of up to 512 tokens [39]. Since user messages were concatenated, relevant information might have been lost in longer texts. In real-world scenarios or early prediction stages with shorter messages, these models may perform better by processing the full text without truncation.

3.5. Approaches for Task 2: Type of Addiction Detection

To properly train and compare our model, we used the same dataset, dividing the trial+train dataset into training (80%) and testing (20%). Additionally, we used the stratify parameter in the train_test_split function to ensure that the class proportion in both sets is similar to the original distribution. This helps maintain a proper class balance, which is especially important when working with imbalanced data like this.

In the following table, we can see how these classes are balanced in the training set.

Table 10
Class Distribution in the Training Set task 2

	users
betting	70
onlinegaming	86
trading	**109**
lootboxes	20

3.5.1. Support Vector Machine

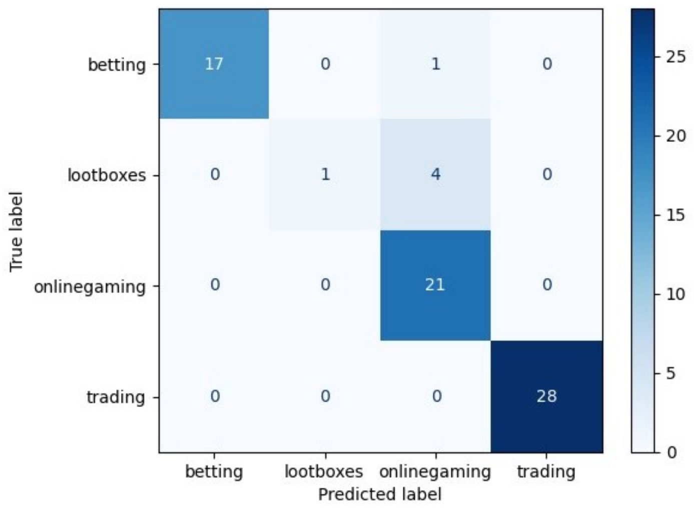
In this case, an SVM model was used again, and to obtain the best model, the same combination of preprocessing techniques and hyperparameters as in task 1 was applied.

If we analyze the confusion matrix of the best model, stopwords_only with the selected hyper-parameters, we can see that the problem lies in the class imbalance, especially with the lootboxes class.

Table 11
Results from SVM with preprocess and hyperparameters task 2

Preprocess	Accuracy	F1-Score	Best Hyperparameters
lemmatization	**0.930556**	0.723404	`C: 10, gamma: scale, kernel: linear, tol: 0.001`
stemming	**0.930556**	0.723404	`C: 10, gamma: scale, kernel: linear, tol: 0.001`
tweet_stemming	**0.930556**	0.723404	`C: 10, gamma: scale, kernel: linear, tol: 0.001`
emoji	0.902778	0.701237	`C: 10, gamma: scale, kernel: linear, tol: 0.001`
stopwords_only	**0.930556**	**0.799595**	`C: 10, gamma: scale, kernel: linear, tol: 0.001`

Figure 3: SVM Confusion Matrix - Task 2 (Best Model: Stopwords Only)



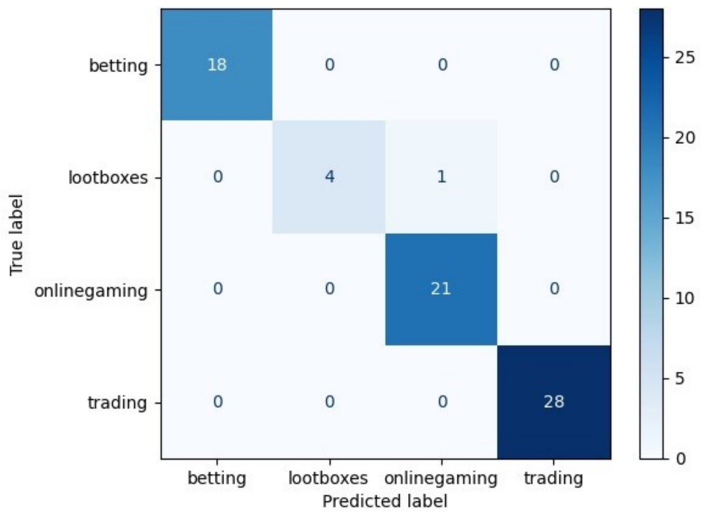
Therefore, it was decided to use the Data Augmentation technique for the lootboxes class. Tests were conducted with different Hugging Face prompting models to generate synthetic data for training. The model that best suited the task was NousResearch/Nous-Hermes-2-Mistral-7B-DPO because it realistically mimics user messages on platforms like Twitch or Telegram, including slang and emoticons. The results improved significantly:

Table 12
Results from SVM with preprocess, DA and hyperparameters task 2

Preprocess	Accuracy	F1-Score	Best Hyperparameters
lemmatization	0.972222	0.926136	`C: 1, gamma: scale, kernel: linear, tol: 0.1`
stemming	0.972222	0.926136	`C: 1, gamma: scale, kernel: linear, tol: 0.1`
tweet_stemming	**0.986111**	**0.966408**	`C: 10, gamma: scale, kernel: linear, tol: 0.001`
emoji	0.958333	0.915672	`C: 1, gamma: scale, kernel: linear, tol: 0.001`
stopwords_only	0.972222	0.926136	`C: 1, gamma: scale, kernel: linear, tol: 0.001`

And as we can see from the confusion matrix, a better result has been achieved for all classes for the new best model.

Figure 4: SVM Confusion Matrix - Task 2 (Best Model: Tweet Stemming)



3.5.2. Transformers Models

The experiments with Transformers used the default hyperparameters, although we applied fine-tuning as detailed below. We also added a TrainerCallback to manage early stopping. Additionally, a T4 x2 GPU from Kaggle was used.

Table 13
Training hyperparameters for transformers in task 2

Hyperparameters	**Value**
Learning Rate	2e-5
Batch Size	16
Seed	100
Max Length	512
Epochs	3
Patience	5
Evaluation Strategy	Steps
Eval Steps	100
Metric for Best Model	eval_loss
Scheduler Type	Linear
Warmup Ratio	0.1

Regarding the transformer model, the same process as in task 1 was carried out, and it was once again demonstrated that the best model for this context is myahan007/bert-base-spanish-wwm-cased-finetuned-tweets. We followed a similar procedure to SVM, initially evaluating the model without data augmentation and later adding DA for the lootboxes class, as in the previous case.

Figure 5: Transformer Confusion Matrix - Task 2

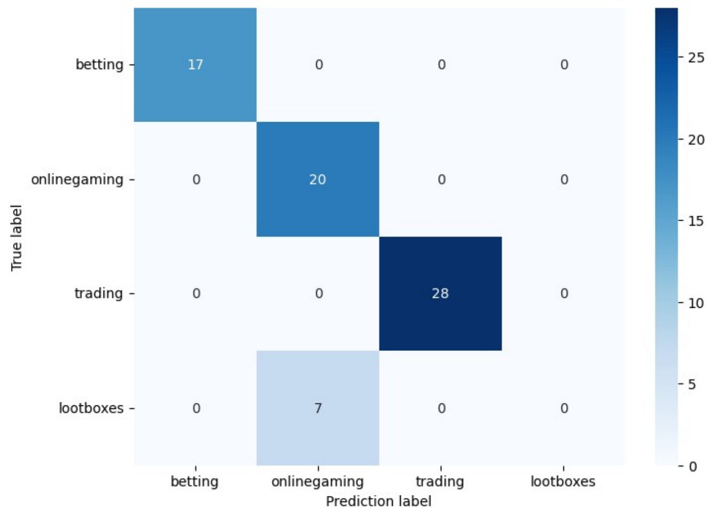
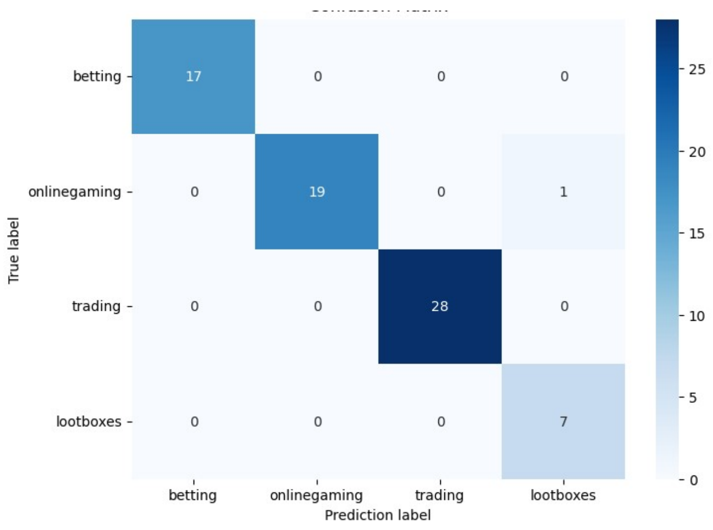
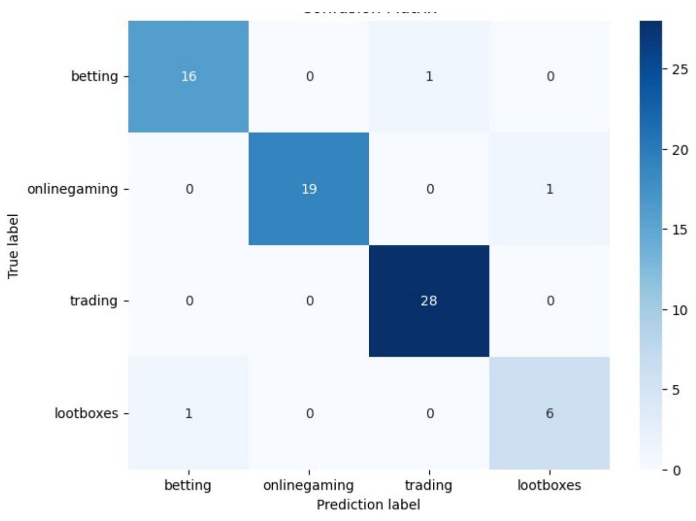


Figure 6: Transformer Confusion Matrix - Task 2 (DA-lootboxes)



Subsequently, we proposed the idea of using the same transformer model but with synthetic data for several classes, so that the training set would be perfectly balanced. This idea emerged to achieve better generalization and to provide more specific information for each class through DA prompting, as previously explained. In this case, we generated 39 instances for betting, 23 for online gaming, and 89 for lootboxes.

Figure 7: Transformer Confusion Matrix - Task 2 (DA-balanced labels)



Finally, we compared the results obtained from the three approaches.

Table 14
Results from transformers for task 2

	Accuracy	F1-Score	Precision	Recall
transformer	0.90278	0.71277	0.68519	0.75
tranformer+DA_lootboxes	**0.98611**	**0.97692**	**0.96875**	**0.9875**
tranformer+DA_balanced	0.95833	0.93878	0.94096	0.93708

Since the competition allows for three runs per task, we decided to choose the best model, which is the DA with lootboxes, along with the last one that balances the classes with synthetic data, aiming to have more messages and better generalization.

4. Runs

Finally, we selected the best models, which are presented in the following table:

Table 15
Summary of approaches for each run

	task	model	Accuracy	F1-Score	Precision	Recall
run0	1	SVM	0.70833	0.70861	0.71130	0.71212
run1	1	SVM+score_enfermo	**0.75000**	**0.75000**	**0.75524**	**0.75524**
run2	1	transformer	0.60824	0.58437	0.59722	0.56707
run0	2	SVM+DA_lootboxes	**0.98611**	0.96640	**0.98863**	0.95000
run1	2	tranformer+DA_lootboxes	**0.98611**	**0.97692**	0.96875	**0.98750**
run2	2	tranformer+DA_balanced	0.95833	0.93878	0.94096	0.93708

For the model selection, we opted for those that demonstrated both strong technical foundations and high performance in the evaluation phase.

For Task 1, we used a Support Vector Machine (SVM) model with the stopwords_only preprocessing strategy, which is particularly well-suited for text classification tasks. In Run 1, we enriched the input features by including a zero-shot prediction score, which can provide an additional advantage by anticipating certain labels in ambiguous cases. Finally, we wanted to include a transformer-based model, known for its powerful capabilities in capturing complex language patterns. This model is also pretrained, which helps improve performance even with limited task-specific data.

For Task 2, we explored the use of SVM for a multi-class classification scenario. We used the tweet_stemming preprocessing strategy and applied data augmentation to the lootboxes class, as it was the most imbalanced. In addition to the classical approach, we reused the best transformer model from Task 1 to assess its generalization capability in a multi-class setting. In Run 1, we applied data augmentation specifically to lootboxes, which yielded excellent results. In Run 2, we balanced all classes through augmentation to enhance the model's ability to generalize across categories.

4.1. Run Configuration

The transformer models were stored in my personal Hugging Face account, while the SVM models were saved directly in my Kaggle notebook. This made them easy to retrieve, as they were already trained and ready for inference.

For the prediction process, each user's message was collected individually and concatenated with the previous one as new messages arrived. This improved efficiency by avoiding the need to reconstruct the full message history repeatedly. Additionally, when using the same preprocessing method, it was not applied more than once, optimizing execution time.

Following the competition's guidelines, carbon emissions were also measured using the CodeCarbon tool. Metrics such as total RAM usage, CPU usage percentage, floating point operations per second (FLOPS), total processing time, and CO_2 emissions in kilograms were recorded. These efficiency metrics help assess the environmental impact and resource demand of the system, providing insight into the solution's suitability for low-resource environments like mobile devices or personal computers.

5. Results

In this section, we present the results obtained in the competition.

5.1. Task 1

Table 16
Results for the competition on Task 1

	Model	Accuracy	Macro_P	Macro_R	Macro_F1
run0	SVM	0.550 (6)	0.632 (3)	0.534 (8)	0.436 (17)
run1	SVM+score_enfermo	0.569 (3)	0.636 (2)	0.554 (4)	0.484 (10)
run2	transformer	0.538 (11)	0.535 (14)	0.534 (9)	0.532 (7)
highest		**0.581**	**0.657**	**0.574**	**0.567**

The results obtained are slightly below our expectations, but considering the best outcomes, our performance was still quite satisfactory.

Despite not reaching the top positions, we achieved the lowest values in the competition in Run 0 with ERDE30 of 0.242, latencyTP of 2, and speed of 0.990, indicating that this model achieves more effective early detection compared to other runs. Similarly, in Run 1, we obtained quite competitive values in the competition with ERDE30 of 0.248, latencyTP of 3, and speed of 0.981. Although the ERDE30 in Run 2 is 0.340 and slightly higher, it is still considered acceptable within the competition.

SVM models stand out in Task 1 of early detection due to their ability for efficient processing, low latency, and high speed (close to 0.990), achieving a low ERDE30. This is due to their use of TF-IDF vectors, allowing for fast predictions even with few messages. Although Transformers are better at capturing context, their limitation of 512 tokens and longer inference time make them less effective for early detection. Additionally, SVM models tend to predict more false positives in the "High Risk" class compared to "Low Risk", which, although increasing some false positives, is useful for early alerts, prioritizing the detection of high risks. This combination of speed, low latency, and focus on high-risk alerts makes SVM models particularly effective for this task.

5.2. Task 2

Table 17
Results for the competition on Task 2

	Model	Accuracy	Macro_P	Macro_R	Macro_F1
run0	SVM+DA_lootboxes	0.931 (2)	0.923 (5)	0.906 (2)	0.912 (2)
run1	tranformer+DA_lootboxes	**0.938 (1)**	**0.952 (1)**	**0.915 (1)**	**0.927 (1)**
run2	tranformer+DA_balanced	0.875 (7)	0.902 (11)	0.823 (7)	0.825 (7)
highest		**0.938**	**0.952**	**0.915**	**0.927**

As we can see, the results for task two are very good, achieving a podium position in two out of the three runs. These first two models perform well because the DA technique was applied correctly, obtaining good synthetic data for the most unbalanced class, lootboxes.

However, we can observe that for run2, the DA technique slightly worsens the performance. This may be because generating a large amount of synthetic data can cause it to deviate somewhat from the reality found in messages from Telegram or Twitch. In other words, in run2, training with more synthetic data slightly affects the performance, and this is due to the fact that the quality of the generated data is not perfect.

6. Conclusion

Participating in the MentalRisk-2025 competition [5] has been an enriching experience that allowed us to apply and consolidate knowledge in natural language processing, machine learning, and transformer-based models. Throughout the project, we achieved solid results by combining classical approaches such as SVM with more advanced techniques, including transformer models and data augmentation strategies.

One of the most relevant aspects of this project was the use of data augmentation (DA) techniques, particularly the generation of synthetic data through prompts using large language models (LLMs). These methods proved especially effective in scenarios with limited training data. The strong performance observed in our models for Task 2 after applying DA techniques confirms their effectiveness in low-resource or imbalanced contexts [40]. This aligns with findings from previous editions of similar competitions, where top-performing systems also leveraged such strategies.

Another key factor in our results was the use of transformer-based models. Despite their limitations in capturing full context in long inputs, these models proved highly effective. Their pretraining on large corpora—many of which exhibit similar characteristics to the dataset used in this competition (informal, abbreviated, conversational language)—allowed for highly valuable knowledge transfer. In fact, our best-performing models in terms of F1-score, both for the binary and multi-class tasks, were based on this architecture.

Among the main limitations of our work, we must highlight the ineffectiveness of data augmentation for Task 1. Consequently, none of the three submissions for this task employed DA. This technique failed to work adequately, likely due to the challenge of capturing the subtle differences between high- and low-risk users—differences which can be quite nuanced and difficult to simulate synthetically.

Another significant limitation, as previously mentioned, lies in the restricted capacity of our models to process long texts. In this competition, message lengths varied, and longer texts posed challenges for our models, making it harder to capture complete context and key information.

Lastly, we did not focus on the computational cost or emissions generated by model training and inference during the competition. These aspects should be considered in future work to improve the sustainability and efficiency of proposed solutions.

Future Work

To address current limitations, explore new techniques, and enhance the models' ability to handle more complex contexts, we have identified several future research directions:

- **Apply Data Augmentation to Task 1:** Explore more targeted text generation techniques that better capture the subtle differences between high- and low-risk users, in order to generate more realistic and useful synthetic data.
- **Evaluate specialized models for long sequences:** Include architectures such as LongFormer [41], which may improve performance on long texts by capturing the user's full message context more effectively.
- **Expand the study of generative models:** Investigate the use of other LLMs to produce higher-quality synthetic data, and explore generation control techniques through more precise prompts.
- **Incorporate explainability techniques:** Integrate interpretability methods such as LIME [42] or SHAP [43] to analyze model behavior and identify the most relevant textual features in classification. This will help avoid black-box decisions and improve the transparency of results.
- **Optimize computational resources:** Explore solutions that strike a balance between performance and efficiency, reducing computational costs without compromising prediction quality.
- **Explore multimodal approaches:** Include other types of data (e.g., metadata, temporal information, or images if available in future editions) to improve predictions. For example, Bucur et al. (2023) propose a multimodal transformer model enriched with temporal information (time2vec), combining text and images to detect depression on social media, achieving better results than text-only models [44].

7. Acknowledgments

Grant PID2023-148577OB-C21 (Human-Centered AI: User-Driven Adapted Language Models-HUMAN_AI) by MICIU/AEI/ 10.13039/501100011033 and by FEDER/UE.

Declaration on Generative AI

During the preparation of this work, the authors used Gemini and Grammarly in order to: check grammar, spelling and reword. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] A. M. Mármol-Romero, P. Álvarez Ojeda, A. Moreno-Muñoz, F. M. Plaza-del Arco, M. D. Molina-González, M. T. Martín-Valdivia, L. A. Ureña-López, A. Montejo-Ráez, Overview of mentalriskes at iberlef 2025: Early detection of mental disorders risk in spanish, Procesamiento del Lenguaje Natural 75 (2025).

[2] World Health Organization, Trastornos mentales, https://www.who.int/es/news-room/fact-sheets/detail/mental-disorders, 2024. Consultado el 18 de mayo de 2025.

[3] Plan Nacional sobre Drogas, 1 de cada 12 jóvenes de 18 a 25 años que participa en apuestas online desarrolla problemas con el juego, https://www.dsca.gob.es/es/comunicacion/notas-prensa/12-jovenes-18-25-anos-participa-apuestas-online-desarrolla-problemas-juego, 2024. Nota de prensa. Consultado el 18 de mayo de 2025.

[4] J. Parapar, P. Martín-Rodilla, D. E. Losada, F. Crestani, Overview of erisk at clef 2021: Early risk prediction on the internet (extended overview)., CLEF (Working Notes) 1 (2021) 864–887.

[5] J. Á. González-Barba, L. Chiruzzo, S. M. Jiménez-Zafra, Overview of IberLEF 2025: Natural Language Processing Challenges for Spanish and other Iberian Languages, in: Proceedings of the

Iberian Languages Evaluation Forum (IberLEF 2025), co-located with the 41st Conference of the Spanish Society for Natural Language Processing (SEPLN 2025), CEUR-WS. org, 2025.

[6] P. Álvarez Ojeda, M. V. Cantero-Romero, A. Semikozova, A. Montejo-Ráez, The precom-sm corpus: Gambling in spanish social media, in: Proceedings of the 31st International Conference on Computational Linguistics, 2025, pp. 17–28.

[7] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (1995) 273–297.

[8] S. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. E. Barnes, D. E. Brown, Twenty years of machine-learning-based text classification: A systematic review, Algorithms 16 (2023) 236. doi:10.3390/a16050236.

[9] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 8440–8451. URL: https://aclanthology.org/2020.acl-main.747/. doi:10.18653/v1/2020.acl-main.747.

[10] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2019).

[11] J. Cañete, G. Chaperon, R. Fuentes, J. Pérez, B. Poblete, Spanish pre-trained bert models and evaluation data, https://github.com/dccuchile/beto, 2020. Accessed May 18, 2025.

[12] M. Bayer, M.-A. Kaufhold, C. Reuter, A survey on data augmentation for text classification, ACM Computing Surveys 55 (2022) 1–39.

[13] S. Lee, L. Liu, W. Choi, Iterative translation-based data augmentation method for text classification tasks, IEEE Access 9 (2021) 160437–160445.

[14] Helsinki-NLP, Opus-MT Spanish-English translation model, 2020. URL: https://huggingface.co/Helsinki-NLP/opus-mt-es-en, accedido el 8 de junio de 2025.

[15] Helsinki-NLP, Opus-MT English-Spanish translation model, 2020. URL: https://huggingface.co/Helsinki-NLP/opus-mt-en-es, accedido el 8 de junio de 2025.

[16] R. Sennrich, B. Haddow, A. Birch, Improving neural machine translation models with monolingual data, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL, 2016, pp. 86–96.

[17] Filipe Almeida, Mistral-7B-Instruct-v0.1-sharded, 2024. URL: https://huggingface.co/filipealmeida/Mistral-7B-Instruct-v0.1-sharded, accedido el 4 de junio de 2025.

[18] NousResearch, Nous-Hermes-2-Mistral-7B-DPO, 2024. URL: https://huggingface.co/NousResearch/Nous-Hermes-2-Mistral-7B-DPO, accedido el 4 de junio de 2025.

[19] J. Zhang, Y. Zhao, M. Saleh, P. J. Liu, Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, in: International Conference on Machine Learning, PMLR, 2020, pp. 11328–11339.

[20] Y. Wang, et al., Large language models are zero-shot data generators, in: International Conference on Learning Representations, 2023.

[21] C. Shorten, T. M. Khoshgoftaar, A survey on image data augmentation for deep learning, Journal of Big Data 6 (2019) 60.

[22] M. Honnibal, I. Montani, S. Van Landeghem, A. Boyd, spaCy: Industrial-strength Natural Language Processing in Python (2020). URL: https://spacy.io. doi:10.5281/zenodo.1212303.

[23] E. Loper, S. Bird, Nltk: The natural language toolkit, arXiv preprint cs/0205028 (2002).

[24] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information processing & management 24 (1988) 513–523.

[25] M. F. Porter, Snowball: A language for stemming algorithms, 2001.

[26] K. Gimpel, N. Schneider, B. O'connor, D. Das, D. P. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, N. A. Smith, Part-of-speech tagging for twitter: Annotation, features, and experiments, in: Proceedings of the 49th annual meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, pp. 42–47.

[27] C. Ceriello, contributors, emoji: Emoji for python, https://github.com/carpedm20/emoji, 2024.

Versión utilizada: 2.11.0.

[28] V. Gallego, Xlm-roberta-large-xnli-anli, 2023. URL: https://huggingface.co/vicgalle/xlm-roberta-large-xnli-anli, accedido el 6 de junio de 2025.

[29] G. Research, google-bert/bert-base-uncased, 2019. URL: https://huggingface.co/google-bert/bert-base-uncased, modelo base BERT uncased en inglés.

[30] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), 2019, pp. 4171–4186.

[31] myahan007, myahan007/bert-base-spanish-wwm-cased-finetuned-tweets, 2021. URL: https://huggingface.co/myahan007/bert-base-spanish-wwm-cased-finetuned-tweets, bERT-base en español, Whole Word Masking, afinado en tweets.

[32] B. Project, bertin-project/bertin-roberta-base-spanish, 2021. URL: https://huggingface.co/bertin-project/bertin-roberta-base-spanish, roBERTa-base adaptado para español.

[33] J. De la Rosa, E. G. Ponferrada, P. Villegas, P. G. d. P. Salas, M. Romero, M. Grandury, Bertin: Efficient pre-training of a spanish language model using perplexity sampling, arXiv preprint arXiv:2207.06814 (2022).

[34] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 (2019).

[35] P. . GOB-ES, Plantl-gob-es/roberta-base-bne, 2021. URL: https://huggingface.co/PlanTL-GOB-ES/roberta-base-bne, roBERTa-base entrenado con corpus de la BNE.

[36] H. Face, distilbert/distilbert-base-multilingual-cased, 2019. URL: https://huggingface.co/distilbert/distilbert-base-multilingual-cased, distilBERT multilingüe, versión cased.

[37] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108 (2019).

[38] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, N. A. Smith, Don't stop pretraining: Adapt language models to domains and tasks, arXiv preprint arXiv:2004.10964 (2020).

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[40] Y. Wang, C. Xu, Q. Sun, H. Hu, C. Tao, X. Geng, D. Jiang, Promda: Prompt-based data augmentation for low-resource nlu tasks, arXiv preprint arXiv:2202.12499 (2022).

[41] I. Beltagy, M. E. Peters, A. Cohan, Longformer: The long-document transformer, arXiv preprint arXiv:2004.05150 (2020).

[42] M. T. Ribeiro, S. Singh, C. Guestrin, " why should i trust you?" explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1135–1144.

[43] W. Zhao, T. Joshi, V. N. Nair, A. Sudjianto, Shap values for explaining cnn-based text classification models, arXiv preprint arXiv:2008.11825 (2020).

[44] A.-M. Bucur, A. Cosma, P. Rosso, L. P. Dinu, It's just a matter of time: Detecting depression with time-enriched multimodal transformers, in: European conference on information retrieval, Springer, 2023, pp. 200–215.