Overview of the CLEF 2024 JOKER Task 2: Humour Classification According to Genre and Technique

Victor Manuel Palma Preciado[1,2], Grigori Sidorov[2], Liana Ermakova[1,*], Anne-Gwenn Bosser[3], Tristan Miller[4,5] and Adam Jatowt[6]

[1]*Université de Bretagne Occidentale, HCTI, France*
[2]*Instituto Politécnico Nacional (IPN), Centro de Investigación en Computación (CIC), Mexico City, Mexico*
[3]*École Nationale d'Ingénieurs de Brest, Lab-STICC CNRS UMR 6285, France*
[4]*Department of Computer Science, University of Manitoba, Winnipeg, Canada*
[5]*Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria*
[6]*University of Innsbruck, Austria*

Abstract

This paper presents details of Task 2 of the JOKER-2024 track, which was held as part of the 15th Conference and Labs of the Evaluation Forum (CLEF 2024). The JOKER-2024 aims to foster progress in different humour processing techniques. In JOKER-2024 Task 2, participants aim to classify sentences in English that use a specific humour technique or genre. In this paper, we present the data used for this task and review the participants' results.

Keywords

humour, humour classification, humour genre, humour technique

1. Introduction

Humour is highly subjective, encompassing a wide array of elements such as emotions, wordplay, and double meanings. Being able to evaluate humour from various perspectives is crucial to obtain insights into the fundamental nature of comedic expression. The wide range of humour techniques and genres presents a significant challenge to such analysis, even in seemingly simple tasks. Although humour is challenging even for humans to analyse, certain features can be used to provide computational assistance, allowing their superior pattern-matching ability to be leveraged. Consequently, automated humour detection methods could generalise some aspects of humour more effectively than the subjective interpretations of humans.

Despite significant advancements in large language models (LLMs) in recent years, creating fully automated systems for humor interpretation, analysis, generation, and translation remains a challenging task. This paper presents Task 2 of the third edition of the JOKER lab at the Conference and Labs of the Evaluation Forum (CLEF) aiming at Humour Classification According to Genre and Technique. Each edition of JOKER develops and releases reusable, quality-controlled datasets for training and testing various humor processing tasks.

This year's JOKER lab has three shared tasks:

Task 1 Humour-aware information retrieval

Task 2 Humour classification according to genre and technique

Task 3 Translation of puns from English to French.

CLEF 2024: Conference and Labs of the Evaluation Forum, September 9–12, 2024, Grenoble, France
*Corresponding author.
0000-0001-8711-1106 (V. M. Palma Preciado); 0000-0003-3901-3522 (G. Sidorov); 0000-0002-7598-7474 (L. Ermakova); 0000-0002-0442-2660 (A. Bosser); 0000-0002-0749-1100 (T. Miller); 0000-0001-7235-0665 (A. Jatowt)

In this paper, we present Task 2 of the JOKER whose main objective is to classify textual humour according to its genre or technique. Task 2 was the most popular JOKER task this year with 18 teams submitting 54 runs over 103 runs submitted to the track in total.

For the general presentation of the JOKER 2024 edition, refer to the overview paper [1]. For the more detailed discussions of Task 1 on retrieving [2] and Task 3 on translating [3] humourous texts, refer to the respecting Task overview papers.

The rest of the paper is organised as follows. Section 2 describes the task, the data collection, and the evaluation metrics. Section 3 provides an overview of the participants' approaches. Section 4 presents and discusses the participants' results on the train and test data as well as the analysis of the results per class. Section 5 concludes the paper.

2. Task Description

In this section, we explain the CLEF 2024 JOKER track's Task 2 on classifying humorous texts.

For the purposes of this task, we constructed a humour taxonomy based on integrating various existing humour classifications and taxonomies used in the literature and in different, separate corpora covering particular aspects of humour. All texts in the corpus were analysed by a professional specialising in humour research, who annotated each text with a human classification label according to the following humour technique classification:

IR: Irony relies on a gap between the literal meaning and the intended meaning, creating a humorous twist or reversal.

SC: Sarcasm involves using irony to mock, criticise, or convey contempt.

EX: Exaggeration involves magnifying or overstating something beyond its normal or realistic proportions.

AID: Incongruity/Absurdity refers (in the case of incongruity) to the unexpected or contradictory elements that are combined in a humorous way and (for absurdity) involve presenting situations, events, or ideas that are inherently illogical, irrational, or nonsensical.

SD: Self-deprecating humour involves making fun of oneself or highlighting one's own flaws, weaknesses, or embarrassing situations in a lighthearted manner.

WS: Wit/Surprise refers (in the case of wit) to clever, quick, and intelligent humour, and (for surprise) to introducing unexpected elements, twists, or punchlines that catch the audience off guard.

Thus, the humour classification of Task 2 is a classification where the goal is to identify in a target text the particular technique used for generating humour.

The data for this task is a mixture of existing corpora on irony and sarcasm detection [4, 5] and on COVID-19 humour [6], our JOKER corpus 2023 [7, 8], and jokes retrieved from public humour sites according to the predefined categories selected in a balanced manner. An example data instance is given below:

Sentence "*Finally figured out the reason I look so bad in photos. It's my face.*"

Humour technique Self-deprecating (SD)

The details on the amount of data for each class in the training and test sets are given in Table 1. The Table provides evidence that the test and the train data come from the same distribution. There are 1,715 sentences in the training set all labelled as either SC, EX, WS, SD, AID, or IR (as discussed above). The test data consists of 6,642 unlabelled texts that contain one of the earlier described types of humour. From these texts, 722 were used for the evaluation.

Runs for the task are evaluated according to standard metrics for classification, namely:

Table 1
Statistics of data per class

Class	# texts		
	test	train	total
Irony (IR)	147	356	503
Sarcasm (SC)	59	162	221
Exaggeration (EX)	106	210	316
Incongruity/Absurdity (AID)	270	634	904
Self-deprecating (SD)	91	228	319
Wit/Surprise (WS)	49	125	174
Total	722	1,715	2,437

- Precision - the ratio of true positive predictions (correctly identified positive instances) to the total number of positive predictions made by the classifier (both true positives and false positives).
- Recall - the ability of a classifier to identify all relevant instances in a dataset. It is the ratio of true positive predictions (correctly identified positive instances) to the total number of actual positive instances (both true positives and false negatives).
- F_1 - the harmonic mean of precision and recall.
- Accuracy - the ratio of the number of correct predictions to the total number of instances.

We report precision, recall, and F_1 scores per class, as well as macro average (averaging the unweighted mean per class) and weighted average (averaging the support-weighted mean per class).

3. Participants' submissions and approaches

In this section, we detail the approaches to humour classification as deployed by participants of the track.

Task 2 proved to be the most popular task at JOKER-2024, with 54 submissions – see Table 2. We observed a great variety of approaches, ranging from state-of-the-art large language models (LLMs) to more classic probabilistic ones. The list below explains the approaches used by each participating team and briefly introduces their work; we also provide citations to the respective system description papers for those teams that submitted them to the workshop proceedings.

The **AB&DPV** team [9] submitted a total of seven runs, opting to use embeddings with the help of Word2Vec. To develop their results, they used the Multilayer Perceptron (MLP), Random Forest, Decision Tree, and Gaussian Naive Bayes classifiers.

The **CodeRangers** team [11] submitted a total of two runs. The team used BERT-uncased and RoBERTa by fine-tuning them on the provided data for Task 2. They observed slightly higher accuracy for RoBERTa than for BERT during their experiments on the training data.

The **CYUT** team [10] submitted a total of three runs. RoBERTa was first fine-tuned, using an 80/20 split of the dataset we shared to train and validate models. GPT-4 was used with zero-shot prompting and chain-of-thought prompting. Attempts to classify among all the classes at once proved too challenging for the model. Therefore hierarchical categories were created and a four-step classification method was employed (using either binary or three-way classification for a given step). After first discriminating AID and WS on the one hand from IR, SC, SD, and EX on the other, further steps were used to classify down the grouping hierarchy. Llama 3-8b was fine-tuned on a single GPU, which was made possible through four-bit quantisation with QLoRa.

The **DadJokers** team [12] submitted a total of three runs. The team's first classification approach is done using BERT base uncased and the second attempt for classification uses a traditional machine learning model, the Random Forest classifier. The authors applied `TFIDFVectorizer` and `SentenceTransformer` as preprocessing steps.

Table 2
Number of submissions per team

Team	# submissions
AB&DPV [9]	7
Arampatzis	8
CYUT [10]	3
CodeRangers [11]	2
DadJokers [12]	3
Dajana&Kathy	1
Frane	1
Jokester [13]	1
HumourInsights [14]	1
NLPalma [15]	3
NaiveNeuron [16]	3
ORPAILLEUR [17]	9
Petra&Regina [18]	1
PunDerstand [19]	4
RubyAiYoungTeam	1
Tomislav&Rowan [20]	3
UAms [21]	1
VayamSolveKurmaha [22]	2
Total	54

The **Dajana&Kathy** team submitted a single run. Their approach involves the use of TF–IDF and BERT embeddings with a variety of models such as SVM, Random Forest, LSTM, and Transformers. A similar approach was taken by the **Frane** team.

The **Jokester** team [13] submitted a single run. They combined several classifiers available through the Scikit library: a voting classifier weighted the results obtained by an SVC and a stack of Random Forest, Decision Tree, Gradient Boosting, and Logistic Regression. We do not report their results as all texts predicted as belonging to the SD class.

The **HumourInsights** team [14] submitted one run. Although this team reported using a variety of classical approaches for the classification task, they chose to submit only their best model. They used TF–IDF to extract features for later use in different methods. They employed boosting methods such as ADA and Gradient with mixed results, but these did not reach the accuracy obtained with KNN and Random Forest.

The **NLPalma** team [15] made three submissions for the classification task using two different approaches: one with more classical classifiers and the other with a more well-known model in the BERT-like lineage.

The **PunDerstand** team [19] submitted a total of four runs. The authors employed the DeBERTa model which, after fine-tuning, gave rise to two runs, one on a raw, unprocessed dataset and one on balanced data. The latter was ensured by an undersampling strategy. In another run, they used GPT-4o, the most recent large language model developed by OpenAI. Few-shot prompting was employed with one example for each class of humour. The team also provided a run with manually guided annotation.

The **Tomislav&Rowan** team [20] submitted a total of three runs. After preprocessing the text, TF–IDF was used for vectorising it. Three models were trained on the data: Logistic Regression, Naïve Bayes, and an SVM.

The **Petra&Regina** team [18] submitted a total of one run. Processed data was vectorised using TF–IDF and class labels were encoded using a linear regression algorithm.

The **UAms** team [21] submitted a single run. This team chose to use a BERT classifier trained on 90% of the training data, with taking special precautions against overfitting.

The **ORPAILLEUR** team [17] submitted a total of nine runs. The team explored the potential of advanced LLMs within a consistent methodological framework. They employed a four-bit quantised

version of three LLMs: Llama2-7b1, Mistral-7b2, and Llama3-8b. The final hidden state of the last token was used as input to the feed-forward layer with the softmax function to get the class probabilities. The team also explored QLoRa adapters.

The **NaiveNeuron** team [16] submitted three runs through iterations of different LLMs, including various instances of GPT-3.5, GPT-4, and GPT-4.0 Plus RAG. They obtained the best results with GPT-4+RAG using a 70/15/15 split for testing. The experimentation of this team was not limited to GPT models; they also used fastText and Llama 3. However, they achieved more favorable results with zero-shot and few-shot classification using GPT-RAG.

The **Arampatzis** team submitted eight runs for this task. The team has experimented with the following approaches: XLNet, Multilayer Perceptron, BERT, RoBERTa, DistilBERT, DeBERTa, Electra, AlBERT.

Finally, the **RubyAiYoungTeam** team submitted a single run, without providing details of their approach.

4. Results

This section details the results of the CLEF 2024 JOKER Task 2 on Humour Classification.

A total of 18 teams submitted 54 runs for Task 2. This was the most popular JOKER task this year which might be explained by the variety of classification models. Participants used mostly LLMs and traditional classifiers although some teams experimented with fine-tuned models and different setups. Some runs had problems with additional classes in their predictions (e.g., "ERROR"). We filtered out these predictions, which explains some differences in the numbers of instances in the runs.

4.1. Test results

Table 4.4 presents the results of the participants on the test data in terms of accuracy, macro and weighted-average precision, recall, and F_1 for each run. We also report the number of instances used for evaluation (#) over 722 test instances. The accuracy of classification varies from 19% to 76%, suggesting that humour classification remains a challenging task. Some runs based on LLAMA 2 outperformed runs based on LLAMA 3. The best results were achieved by the ORPAILLEUR team with the Mistral model.

Table 4 presents the results for each run and each class; we report precision, recall, and F_1. The most difficult classes were Exaggeration and Wit/Surprise. The latter category is a combination of two types of humour which might be a reason for its difficulty. Further analysis is needed.

4.2. Train results

To check for a possible overfitting problem, we report training data results for each run in Tables 5 and 6. Table 5 presents the results the training data in terms of accuracy, macro and weighted-average precision, recall, and F_1, while Table 6 presents the results for each run and each class in terms of precision, recall, and F_1. Random Forest, LLAMA, and Mistral models achieved 100% accuracy on the training data, while on the test data, Random Forest did not score well, suggesting the existence of an overfitting problem. GPT-4 obtained very low scores both on test and training data. However, LLMs showed much better generalisation capacity on unseen data than traditional models.

We conducted various analyses to better understand the peculiarities of the dataset and the results of each team. The following two subsections present some figures and discussion focussing on performance across classes of data and across the various system archetypes used by the participants.

Table 3
Accuracy; macro-average precision, recall, and F_1; weighted-average precision, recall, and F_1; and number of instances attempted (test data) - reported in % (sorted on overall accuracy)

		macro average			weighted average			
Run ID	A	P	R	F_1	P	R	F_1	#
ORPAILLEUR_mistral-7b-ens	76	71	70	70	75	76	75	722
ORPAILLEUR_llama2-ens	74	68	67	66	74	74	72	722
ORPAILLEUR_llama3-8b-ens	73	67	65	66	72	73	72	722
ORPAILLEUR_mistral-7b-high	72	67	66	66	72	72	72	722
ORPAILLEUR_llama2-high	71	63	65	64	71	71	71	722
Code Rangers_roberta	70	75	63	59	78	70	66	509
ORPAILLEUR_llama2-low	70	65	64	62	71	70	68	722
ORPAILLEUR_llama3-8b-high	70	63	61	61	69	70	69	722
ORPAILLEUR_llama3-8b-low	70	64	63	63	70	70	70	722
CYUT_llama3-fine-tuning	70	64	65	64	70	70	70	718
PunDerstand_DeBERTa	69	59	65	60	68	69	67	722
Arampatzis_BERT	68	60	60	59	67	68	67	722
Arampatzis_deberta	68	61	62	61	67	68	67	722
PunDerstand_DeBERTaSampled	68	60	65	62	69	68	68	722
Arampatzis_DistilBertTokenizer	68	61	58	59	66	68	66	722
DadJokers_bert_base_uncased	67	60	60	60	67	67	67	722
ORPAILLEUR_mistral-7b-low	67	65	61	60	69	67	66	722
NLPalma_BERTd	67	60	60	59	67	67	67	722
PunDerstand_GuidedAnnotation	67	65	64	61	73	67	67	45
CodingRangers_bert_uncased	67	59	61	59	66	67	66	722
Arampatzis_Roberta	66	58	57	56	64	66	65	722
Demonteam_BERTM	66	58	58	58	65	66	65	722
Arampatzis_MLP	66	58	57	57	65	66	65	722
Arampatzis_XLNet	65	57	57	57	63	65	64	722
Arampatzis_Albert	65	58	60	58	65	65	64	722
UAms_BERT_ft	63	57	58	52	66	63	60	722
VayamSolveKurmaha_BERT	60	54	53	51	59	60	58	722
NLPalma_PREDCNN	60	54	50	51	58	60	59	722
Arampatzis_electra	60	32	41	35	47	60	51	722
NaiveNeuron_fastText	59	51	51	51	58	59	58	722
VayamSolveKurmaha_BERT	57	50	53	50	59	57	57	722
DadJokers_RandomForest_MLP_Ensemble	56	49	48	47	54	56	53	722
HumourInsights_Random Forest	55	50	45	45	53	55	52	722
UBO_RubyAiYoungTeam	53	53	39	40	52	53	48	722
team1_Petra_and_Regina_LogisticRegression	53	53	39	40	52	53	48	722
Dajana&Kathy_Joker_LogisticRegression	53	53	39	40	52	53	48	722
team1_FRANE_AND_ANDREA_LogisticRegression	53	53	39	40	52	53	48	722
NaiveNeuron_llama3:70b_rag	53	50	53	50	57	53	54	722
NaiveNeuron_llama3:70b_rag-uae	53	50	53	50	57	53	53	722
DadJokers_RandomForest	52	54	37	37	53	52	46	722
Tomislav&Rowan_SVM	51	44	37	38	48	51	47	722
Tomislav&Rowan_LogisticRegression	48	42	31	31	45	48	41	722
AB&DPV_MLP3000params	48	41	38	38	45	48	44	722
PunDerstand_GPT4oFewShot	47	43	46	42	53	47	47	722
Tomislav&Rowan_NaiveBayes	44	29	23	19	36	44	32	722
AB&DPV_RandomForestClassifier250	38	36	21	19	38	38	29	722
AB&DPV_RandomForestClassifier500	38	34	21	19	36	38	29	722
AB&DPV_MLP2000	37	09	17	10	15	37	21	722
AB&DPV_MLP3000	37	09	17	10	15	37	21	722
CYUT_GPT-4	36	39	40	34	42	36	32	591
AB&DPV_DecisionTreeClassifier	29	23	22	22	29	29	28	722
AB&DPV_GaussianNB	27	20	24	17	29	27	25	722
CYUT_roBERTa-fine-tuning	19	19	24	21	17	19	17	722

Table 4
Precision, recall, and F_1 per class (test data) - reported in % (sorted on overall accuracy)

	SD			WS			EX			IR			SC			AID		
Run ID	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
ORPAILLEUR_mistral-7b-ens	76	78	77	61	51	56	64	43	52	67	83	74	73	75	74	87	88	87
ORPAILLEUR_llama2-ens	77	82	80	55	45	49	64	25	36	60	78	68	65	81	72	89	91	90
ORPAILLEUR_llama3-8b-ens	76	85	80	54	39	45	48	41	44	59	72	65	76	66	71	90	89	90
ORPAILLEUR_mistral-7b-high	71	75	73	53	41	46	58	47	52	65	77	70	69	69	69	86	85	85
ORPAILLEUR_llama2-high	76	81	78	45	37	40	51	49	50	67	59	63	52	73	61	89	89	89
Code Rangers_roberta	37	66	47	58	87	70	85	64	73	100	1	3	86	65	74	83	93	88
ORPAILLEUR_llama2-low	82	71	76	46	43	44	63	21	31	55	74	63	57	83	68	86	89	87
ORPAILLEUR_llama3-8b-high	75	85	79	41	24	31	45	37	40	56	67	61	73	64	68	87	90	88
ORPAILLEUR_llama3-8b-low	77	79	78	46	37	41	49	39	43	54	65	59	68	68	68	88	88	88
CYUT_llama3-fine-tuning	70	69	70	44	63	52	52	41	46	63	60	62	67	68	67	86	88	87
PunDerstand_DeBERTa	68	90	77	45	51	48	43	19	26	59	60	60	50	83	62	92	86	89
Arampatzis_BERT	74	82	78	44	31	36	50	29	37	55	69	62	51	59	55	86	87	86
Arampatzis_deberta	75	84	79	41	55	47	48	25	33	58	54	56	58	66	62	83	90	86
PunDerstand_DeBERTaSampled	68	90	78	44	57	50	41	43	42	66	42	51	52	71	60	91	85	88
Arampatzis_DistilBertTokenizer	72	84	78	52	31	38	42	35	38	64	56	59	57	54	56	79	92	85
DadJokers_bert_base_uncased	71	79	75	46	35	40	42	39	40	58	59	59	59	64	62	85	86	85
ORPAILLEUR_mistral-7b-low	67	84	74	61	47	53	53	22	31	49	82	61	71	51	59	89	79	84
NLPalma_BERTd	63	82	71	46	35	40	49	49	49	66	53	59	50	59	54	83	84	84
PunDerstand_GuidedAnnotation	75	60	67	57	67	62	50	14	22	27	75	40	1	82	90	83	83	83
CodingRangers_bert_uncased	69	84	76	40	51	45	41	28	34	62	57	59	57	61	59	83	85	84
Arampatzis_Roberta	72	78	75	47	31	37	43	21	28	54	62	58	48	63	54	83	90	87
Demonteam_BERTM	68	76	72	48	45	46	44	28	34	57	57	57	52	56	54	81	89	85
Arampatzis_MLP	69	85	76	44	31	36	42	23	29	52	70	60	58	53	55	85	84	85
Arampatzis_XLNet	73	76	75	45	35	39	41	28	33	62	54	58	46	61	53	77	88	82
Arampatzis_Albert	76	78	77	39	61	48	40	27	32	48	53	50	56	54	55	87	84	85
UAms_BERT_ft	68	85	75	31	65	42	60	3	5	55	64	59	45	53	48	85	81	83
VayamSolveKurmaha_BERT	58	38	46	53	37	43	41	18	25	52	63	57	45	76	57	74	84	78
NLPalma_PREDCNN	66	64	65	46	22	30	44	36	40	52	53	53	49	44	46	69	83	75
Arampatzis_electra	74	69	72	0	0	0	0	0	0	39	88	54	0	0	0	78	89	83
NaiveNeuron_fastText	63	75	68	37	37	37	46	37	41	45	48	46	39	31	34	76	79	78
VayamSolveKurmaha_BERT	57	57	57	44	41	43	36	20	26	48	56	52	34	69	46	83	73	78
DadJokers_RandomForest_MLP_Ensemble	59	62	60	41	35	38	31	10	15	45	64	53	47	44	46	69	73	71
HumourInsights_Random Forest	59	68	63	43	41	42	50	12	20	47	44	45	37	24	29	61	83	70
UBO_RubyAiYoungTeam	64	45	53	57	16	25	41	8	14	45	58	51	52	20	29	56	85	68
team1_Petra_and_Regina_LogisticRegression	64	45	53	57	16	25	41	8	14	45	58	51	52	20	29	56	85	68
Dajana&Kathy_Joker_LogisticRegression	64	45	53	57	16	25	41	8	14	45	58	51	52	20	29	56	85	68
team1_FRANE_AND_ANDREA_LogisticRegression	64	45	53	57	16	25	41	8	14	45	58	51	52	20	29	56	85	68
NaiveNeuron_llama3:70b_rag	41	49	45	33	59	42	34	46	39	59	35	44	58	63	60	74	64	69
NaiveNeuron_llama3:70b_rag-uae	41	52	46	37	65	47	35	50	41	59	33	43	57	58	57	74	61	67
DadJokers_RandomForest	66	49	57	46	24	32	50	3	5	46	48	47	60	10	17	52	90	66
Tomislav&Rowan_SVM	54	29	37	46	33	38	33	12	18	42	51	46	28	15	20	59	84	69
Tomislav&Rowan_LogisticRegression	58	23	33	33	12	18	27	6	9	42	41	41	44	12	19	51	91	65
AB&DPV_MLP3000params	49	49	49	39	24	30	30	7	11	42	45	43	33	27	30	53	73	62
PunDerstand_GPT4oFewShot	22	27	24	26	29	27	30	42	35	67	31	42	44	86	58	71	59	64
Tomislav&Rowan_NaiveBayes	90	10	18	0	0	0	0	0	0	40	31	35	0	0	0	44	96	60
AB&DPV_RandomForestClassifier250	63	11	19	18	6	9	40	4	7	26	17	20	29	3	6	40	86	55
AB&DPV_RandomForestClassifier500	57	9	15	27	8	13	30	3	5	27	18	22	25	2	3	40	86	54
AB&DPV_MLP2000	0	0	0	17	2	4	0	0	0	0	0	0	0	0	0	38	99	54
AB&DPV_MLP3000	0	0	0	17	2	4	0	0	0	0	0	0	0	0	0	38	99	54
CYUT_GPT-4	9	9	9	23	69	35	41	21	28	63	12	20	49	68	57	47	64	54
AB&DPV_DecisionTreeClassifier	21	24	22	17	10	13	22	11	15	24	29	26	12	17	14	43	42	42
AB&DPV_GaussianNB	0	0	0	17	27	21	23	7	10	17	3	5	10	56	17	53	50	52
CYUT_roBERTa-fine-tuning	0	0	0	3	6	4	3	8	4	59	54	56	51	73	60	0	0	0

Table 5
Accuracy; macro-average precision, recall, and F_1; weighted-average precision, recall, and F_1; and number of instances attempted (training data) - reported in %

		macro average			weighted average			
Run ID	Acc	P	R	F_1	P	R	F_1	#
DadJokers_RandomForest	100	100	100	100	100	100	100	1715
DadJokers_RandomForest_MLP_Ensemble	100	100	100	100	100	100	100	1715
ORPAILLEUR_llama3-8b-ens	100	99	100	100	100	100	100	1715
ORPAILLEUR_llama2-ens	100	99	99	99	100	100	100	1715
ORPAILLEUR_mistral-7b-ens	100	99	100	99	100	100	100	1715
NaiveNeuron_llama3:70b_rag	98	98	98	98	98	98	98	1715
NaiveNeuron_llama3:70b_rag-uae	97	97	97	97	98	97	97	1715
HumourInsights_Random Forest	97	97	98	98	97	97	97	1715
ORPAILLEUR_llama2-high	95	93	93	93	95	95	95	1715
ORPAILLEUR_llama3-8b-high	94	94	93	93	94	94	94	1715
ORPAILLEUR_mistral-7b-high	94	93	92	93	94	94	94	1715
DadJokers_bert_base_uncased	94	93	92	93	94	94	94	1715
ORPAILLEUR_llama3-8b-low	94	92	92	92	94	94	94	1715
AB&DPV_RandomForestClassifier250	94	97	92	94	94	94	94	1715
AB&DPV_RandomForestClassifier500	94	97	92	94	94	94	94	1715
ORPAILLEUR_llama2-low	94	93	92	92	94	94	94	1715
NaiveNeuron_fastText	93	92	92	92	93	93	93	1715
Tomislav&Rowan_SVM	93	95	90	92	94	93	93	1715
ORPAILLEUR_mistral-7b-low	92	92	90	91	93	92	92	1715
NLPalma_PREDCNN	91	90	88	89	91	91	91	1715
CYUT_llama3-fine-tuning	91	89	89	89	91	91	91	1715
Demonteam_BERTM	90	88	87	88	90	90	90	1715
NLPalma_BERTd	90	87	88	88	90	90	90	1715
CodingRangers_bert_uncased	88	84	85	84	88	88	88	1715
Arampatzis_MLP-test_predictions_bert	87	83	83	83	87	87	87	1715
Arampatzis_DistilBertTokenizer_test_predictions	87	84	84	84	87	87	87	1715
Arampatzis_BERT_test_predictions1	87	83	82	82	87	87	87	1715
Arampatzis_test_predictions_deberta	84	80	80	79	85	84	84	1715
Arampatzis_XLNet_test_predictions	83	78	78	78	82	83	82	1715
Dajana&Kathy_Joker_LogisticRegression	82	90	72	78	85	82	81	1715
team1_FRANE_AND_ANDREA_LogisticRegression	82	90	72	78	85	82	81	1715
team1_Petra_and_Regina_LogisticRegression	82	90	72	78	85	82	81	1715
UBO_RubyAiYoungTeam	82	90	72	78	85	82	81	1715
Tomislav&Rowan_LogisticRegression	80	91	71	77	85	80	79	1715
Arampatzis_Roberta_test_predictions	80	76	73	73	79	80	79	1715
PunDerstand_DeBERTa	80	71	74	71	80	80	79	1715
PunDerstand_DeBERTaSampled	79	73	78	75	81	79	79	1715
AB&DPV_DecisionTreeClassifier	77	73	72	72	77	77	77	1715
Arampatzis_Albert_test_predictions	77	71	72	71	77	77	77	1715
VayamSolveKurmaha_BERT	73	71	67	66	75	73	72	1715
UAms_BERT_ft	73	70	68	62	76	73	70	1715
PunDerstand_GuidedAnnotation	71	61	69	62	74	71	71	87
Code Rangers_roberta	70	57	60	57	64	70	66	1196
VayamSolveKurmaha_BERT	69	64	65	63	71	69	69	1715
Arampatzis_test_predictions_electra	64	51	43	38	60	64	55	1715
Tomislav&Rowan_NaiveBayes	64	88	45	48	78	64	58	1715
AB&DPV_MLP3000params	57	53	49	50	56	57	55	1715
PunDerstand_GPT4oFewShot	45	44	44	41	53	45	46	1715
AB&DPV_MLP2000	37	12	17	11	17	37	21	1715
AB&DPV_MLP3000	37	12	17	11	17	37	21	1715
CYUT_GPT-4	33	35	37	31	39	33	30	1409
CYUT_roBERTa-fine-tuning	26	27	29	28	25	26	25	1715
AB&DPV_GaussianNB	19	29	25	16	35	19	17	1715

Table 6
Precision, recall, and F_1 per class (training data) - reported in %

	SD			WS			EX			IR			SC			AID		
Run ID	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DadJokers_RandomForest	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
DadJokers_RandomForest_MLP_Ensemble	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
ORPAILLEUR_llama3-8b-ens	100	100	100	98	99	98	100	100	100	100	99	100	100	100	100	100	100	100
ORPAILLEUR_llama2-ens	100	100	100	98	98	98	100	99	100	99	99	99	100	100	100	100	100	100
ORPAILLEUR_mistral-7b-ens	100	100	100	98	100	99	100	99	99	100	99	99	100	100	100	100	100	100
NaiveNeuron_llama3:70b_rag	97	100	98	96	96	96	98	99	98	99	98	98	99	100	100	99	98	98
NaiveNeuron_llama3:70b_rag-uae	94	99	97	94	94	94	96	99	97	98	96	97	99	100	99	99	98	98
HumourInsights_Random Forest	95	99	97	98	100	99	100	98	99	96	96	96	96	99	98	98	96	97
ORPAILLEUR_llama2-high	94	95	95	93	89	91	90	91	91	94	92	93	91	95	93	97	98	98
ORPAILLEUR_llama3-8b-high	94	96	95	91	86	88	90	90	90	91	93	92	97	92	94	98	98	98
ORPAILLEUR_mistral-7b-high	96	96	96	92	86	89	88	89	88	91	93	92	93	93	93	98	98	98
DadJokers_bert_base_uncased	95	97	96	92	88	90	87	86	87	92	92	92	93	94	93	97	98	97
ORPAILLEUR_llama3-8b-low	96	95	95	90	89	90	89	86	87	90	93	92	92	94	93	98	97	98
AB&DPV_RandomForestClassifier250	99	92	95	98	86	91	98	90	94	95	92	93	100	93	96	88	99	93
AB&DPV_RandomForestClassifier500	100	91	95	99	86	92	99	90	94	94	92	93	100	93	96	88	99	93
ORPAILLEUR_llama2-low	97	91	94	84	89	86	96	82	88	88	96	92	93	98	95	97	97	97
NaiveNeuron_fastText	94	95	94	92	90	91	87	90	88	92	91	92	94	92	93	96	96	96
Tomislav&Rowan_SVM	99	82	90	98	92	95	98	90	94	88	98	93	98	80	88	91	99	95
ORPAILLEUR_mistral-7b-low	92	96	94	88	86	87	93	81	87	82	95	88	97	89	93	99	96	97
NLPalma_PREDCNN	92	89	91	91	78	84	83	87	85	90	87	89	92	91	91	93	97	95
CYUT_llama3-fine-tuning	91	89	90	74	92	82	86	79	82	90	88	89	94	92	93	95	96	96
Demonteam_BERTM	93	93	93	81	83	82	86	79	82	87	87	87	85	86	86	94	97	95
NLPalma_BERTd	84	94	89	82	82	82	82	80	81	91	85	88	89	93	91	96	95	95
CodingRangers_bert_uncased	94	90	92	68	78	73	72	68	70	88	84	86	88	90	89	95	98	97
Arampatzis_MLP-test_predictions_bert	89	94	91	70	61	65	75	63	69	81	89	85	88	93	90	96	96	96
Arampatzis_DistilBertTokenizer	89	94	91	79	74	77	67	74	70	87	78	82	84	85	84	96	97	96
Arampatzis_BERT_test_predictions1	87	93	90	73	58	65	76	65	70	79	90	84	88	90	89	98	95	96
Arampatzis_test_predictions_deberta	86	88	87	51	70	59	79	46	58	82	85	83	86	94	90	94	96	95
Arampatzis_XLNet_test_predictions	88	87	88	68	63	65	63	56	59	80	73	76	79	93	85	91	97	94
Dajana&Kathy_Joker_LogisticRegression	93	72	81	98	51	67	97	58	73	80	90	85	96	65	77	74	98	85
team1_FRANE&ANDREA_LogisticRegression	93	72	81	98	51	67	97	58	73	80	90	85	96	65	77	74	98	85
team1_Petra_and_Regina_LogisticRegression	93	72	81	98	51	67	97	58	73	80	90	85	96	65	77	74	98	85
UBO_RubyAiYoungTeam	93	72	81	98	51	67	97	58	73	80	90	85	96	65	77	74	98	85
Tomislav&Rowan_LogisticRegression	99	60	74	98	64	77	98	59	74	84	91	87	99	50	66	70	99	82
Arampatzis_Roberta_test_predictions	87	90	89	71	51	60	63	32	42	67	81	73	72	88	79	93	96	94
PunDerstand_DeBERTa	86	94	90	48	57	52	51	33	40	82	76	79	59	88	70	98	95	97
PunDerstand_DeBERTaSampled	77	97	86	53	70	60	53	63	57	84	66	74	75	90	82	98	83	90
AB&DPV_DecisionTreeClassifier	70	77	73	64	50	57	68	71	69	79	79	79	71	67	69	86	86	86
Arampatzis_Albert_test_predictions	80	86	83	46	69	55	51	39	44	67	69	68	86	80	83	95	90	92
VayamSolveKurmaha_BERT	87	48	62	68	64	66	69	30	42	66	76	71	55	93	69	84	92	88
UAms_BERT_ft	77	89	82	40	81	54	75	1	3	63	76	69	70	73	72	93	88	90
PunDerstand_GuidedAnnotation	67	67	67	33	100	50	20	13	15	68	76	72	100	77	87	80	80	80
Code Rangers_roberta	63	79	70	0	0	0	32	57	41	82	91	86	76	62	68	91	71	80
VayamSolveKurmaha_BERT	76	62	68	61	65	63	55	29	38	60	65	63	43	88	57	91	82	86
Arampatzis_test_predictions_electra	81	71	76	0	0	0	0	0	0	42	94	58	100	2	4	84	94	89
Tomislav&Rowan_NaiveBayes	100	29	44	100	16	28	100	30	47	74	82	78	100	12	22	55	100	71
AB&DPV_MLP3000params	68	65	67	47	36	41	38	16	22	47	53	50	57	43	49	63	79	70
PunDerstand_GPT4oFewShot	19	23	21	25	31	28	25	40	31	79	29	42	48	83	61	66	57	61
AB&DPV_MLP2000	0	0	0	12	2	4	21	3	5	0	0	0	0	0	0	37	98	54
AB&DPV_MLP3000	0	0	0	12	2	4	21	3	5	0	0	0	0	0	0	37	98	54
CYUT_GPT-4	6	5	6	22	63	32	27	13	18	63	12	19	53	72	61	41	58	48
CYUT_roBERTa-fine-tuning	0	0	0	1	2	2	0	0	0	82	81	82	81	89	85	0	0	0
AB&DPV_GaussianNB	57	2	3	16	45	24	18	15	17	20	8	11	12	64	20	51	15	23

Table 7
Average precision, recall, and F-score by class on the test set - reported in %

ClassID	Class	Average over runs			# texts
		Precision	Recall	F_1	
SD	Self-deprecating	47	44	44	91
WS	Wit/Surprise	33	26	27	49
EX	Exaggeration	32	18	21	106
IR	Irony	40	41	40	147
SC	Sarcasm	38	38	35	59
AID	Incongruity/Absurdity	56	66	60	270

Table 8
Best, worst, median, and average performance (precision, recall, and F-score) for the Self-deprecating class

SD: Self-deprecating	Precision	Recall	F_1
ORPAILLEUR_llama3-8b-ens	76	85	80
Median_SD	66	69	68
Average_SD	60	58	56
AB	DPV_task_2_RandomForestClassifier500	57	9
15			

Table 9
Best, worst, median, and average performance (precision, recall, and F-score) for the Wit/Surprise class

WS: Wit/Surprise	Precision	Recall	F_1
ORPAILLEUR_mistral-7b-ens	61	51	56
Median_WS	44	35	38
Average_WS	40	32	33
CYUT_roBERTa-fine-tuning	3	6	4

4.3. Performance per class

As was shown already in Table 1 above, the dataset is unbalanced with respect to its class distribution. Therefore, the relevant model performance metrics for this type of dataset tend to be macro and weighted averages.

From Table 7 it can be observed that the classes with the best performance over all submissions are generally those that occur most frequently in the test set. Overall, the best average classification across different runs was for class AID, and the worst, despite being the third largest in terms of number of texts, was class EX.

The following five subsections examine system performance on the individual classes, with tables presenting the best, worst, median, and mean ("average") system performance in terms of precision, recall, and F-score. We excluded four incomplete runs from the computation of the average and median values. All scores are reported in percentage.

4.3.1. SD: Self-deprecating humour

From Table 8 we can observe a very noticeable difference between the median and the highest result for the SD class.

4.3.2. WS: Wit/Surprise humour

For Wit/Surprise (see Table 9), there isn't such a pronounced difference between the median, average, and the best result, although we note that the LLMs lead with the best results. It's worth noting that

Table 10
Best, worst, median, and average performance (precision, recall, and F-score) for the Exaggeration class

EX: Exaggeration	Precision	Recall	F_1
ORPAILLEUR_mistral-7b-high	58	47	52
Median_EX	41	21	29
Average_EX	39	22	26
CYUT_roBERTa-fine-tuning	3	8	4

Table 11
Best, worst, median, and average performance (precision, recall, and F-score) for the Irony class

IR: Irony	Precision	Recall	F_1
ORPAILLEUR_mistral-7b-ens	67	83	74
Median_IR	52	56	53
Average_IR	49	52	49
AB&DPV_GaussianNB	17	3	5

Table 12
Best, worst, median, and average performance (precision, recall, and F-score) for the Sarcasm class

SC: Sarcasm	Precision	Recall	F_1
ORPAILLEUR_mistral-7b-ens	73	75	74
Median_SC	51	56	54
Average_SC	46	47	44
AB&DPV_RandomForestClassifier500	25	2	3

the other models on average are not far behind. Where there is a very marked disparity is in the lowest performance compared to the others. In this case, the BERT-like model was not able to understand this type of class well. Of course, this doesn't mean that BERT-like models are incapable of competing, as the table only shows the worst result. It's clear that other models can perform quite well, especially considering that the median is 42%.

4.3.3. EX: Exaggeration humour

Exaggeration (see Table 10) is a class that shows a very similar trend to the previous ones, with a rather high disparity between the worst and the best performance. It is possible that these results are linked to the way the models' learning was handled.

4.3.4. IR, SC: Irony and Sarcasm humour

The cases of Irony and Sarcasm (see Tables 11 and 12) are peculiar. Perhaps owing to their semantic similarity, both classes show a similar pattern of results, with quite close medians and averages. Furthermore, the fact that the model with the best performance in both is an LLM (specifically, Mistral) supports the argument that LLMs outperform more traditional models. It is worth noting that the ORPAILLEUR team's focus on using LLMs was quite rewarding, as they proved superior in all cases.

4.3.5. AID: Incongruity/Absurdity

The results for Incongruity/Absurdity (see Table 13) are by far the best among all the classes, with even the classical methods achieving good results in the worst case. As expected, the LLMs again lead with a very high score of 90% in F-score.

Table 13
Best, worst, median, and average performance (precision, recall, and F-score) for the Incongruity/Absurdity class

AID: Incongruity/Absurdity	Precision	Recall	F_1
ORPAILLEUR_llama2-ens	89	91	90
Median_AID	77	85	78
Average_AID	70	81	71
AB&DPV_DecisionTreeClassifier	42	42	42

Table 14
Best and worst performance (macro precision, recall, and F1 score) by type of model

Model type	Performance	Team (model)	Macro Average		
			Precision	Recall	F_1
LLM	Best	ORPAILLEUR (Mistral-7b)	71	70	70
BERT-like	Best	PunDerstand (DeBERTa)	59	65	60
Deep learning	Best	NLPALMA (CNN)	54	50	51
Machine learning	Best	HumourInsights (Random Forest)	50	45	45
Probabilistic	Best	Tomislav&Rowan (NaiveBayes)	29	23	19
LLM	Worst	NAIVENEURON (LLaMA3)	50	53	50
BERT-like	Worst	CYUT (RoBERTa)	19	24	21
Deep learning	Worst	AB&DPV (MLP)	9	17	10
Machine learning	Worst	AB&DPV (DecisionTree)	23	22	22
Probabilistic	Worst	AB&DPV (GaussianNB)	20	24	17

4.4. Performance by Model

Tables 14 present the best and worst results for each model type (i.e., the general classes of methods employed by the participants). This is matching the macro averages in Table . We distinguish traditional machine learning models, with special attention to probilistic models. We also distinguish earlier pre-transformer deep learning models, from BERT style (encoder-only) models, and more recent LLMs (decoder-only) models.

We make a number of observations. First, we observe that traditional machine learning models perform well on this multi-class problem, but deep learning models perform better, and modern transformers perform best. Second, it is reassuring that the best or better performing models per class correspond closely to the evolution of text classification models over time, and their expected performance based on other text classification tasks. Third, it is important to note that the performance gain also results in far higher computational costs and less transparency. Modern models such as encoder-only BERT or decoder-only LLMs have very many parameters, and high training and inference costs. In practice, the Joker Task 2 corpus is relatively small, and contains sentence length texts, making the use of this models feasible. Compared to traditional machine learning models like decision trees, they are also less transparent in how they arrive at a label prediction. Fourth, it is interesting to speculate whether the superior language understanding capabilities of the largest models allow them to capture the intertextual aspects and cultural references of the humorous texts, and whether this can lead to further improvement of this complex text classification task.

5. Discussion and Conclusions

This paper provides an overview of Task 2 from the CLEF 2024 JOKER track aiming at classifying humorous texts based on specific humour techniques or genres. We constructed a reusable test collection of 2,437 manually annotated short humouristic texts. A total of 18 teams submitted 54 runs to the JOKER Task 2 on humour classification being the most popular JOKER task this year.

In our evaluation campaign, we observed the best results from the ORPAILLEUR [17] team, whose use of LLMs obtained the highest overall accuracy. We generally found state-of-the-art LLM-based models to clearly outperform traditional classifiers. However, the overall and per-class scores of all participants show significant room for improvement. These results indicate that the semantics and pragmatics of humour is still challenging for LLMs despite their significant recent advances. Our evaluation setup itself could also be improved in terms of the quality and quantity of data, or perhaps of providing individual classification tasks for each technique and genre; these are options that we are exploring for future iterations of the JOKER track.

Acknowledgments

This project has received a government grant managed by the National Research Agency under the program "Investissements d'avenir" integrated into France 2030, with the Reference ANR-19-GURE-0001. This Lab would not have been possible without the great support of numerous individuals; we would like to thank in particular Jaap Kamps for his valuable comments and suggestions.

References

[1] L. Ermakova, T. Miller, A. Bosser, V. M. Palma-Preciado, G. Sidorov, A. Jatowt, Overview of CLEF 2024 JOKER track: Automatic humor analysis, in: L. Goeuriot, G. Q. Philippe Mulhem, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, 2024.

[2] L. Ermakova, A.-G. Bosser, T. Miller, A. Jatowt, Overview of the CLEF 2024 JOKER Task 1: Humour-aware information retrieval, in: G. Faggioli, N. Ferro, P. Galuscakova, A. G. Seco de Herrera (Eds.), Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024), CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[3] L. Ermakova, A.-G. Bosser, T. Miller, A. Jatowt, Overview of the CLEF 2024 JOKER Task 3: Translation of puns from English to French, in: G. Faggioli, N. Ferro, P. Galuscakova, A. G. Seco de Herrera (Eds.), Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2024), CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[4] I. Abu Farha, S. V. Oprea, S. Wilson, W. Magdy, SemEval-2022 Task 6: iSarcasmEval, intended sarcasm detection in English and Arabic, in: Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Association for Computational Linguistics, 2022, pp. 802–814. doi:10.18653/v1/2022.semeval-1.111.

[5] S. Frenda, A. Pedrani, V. Basile, S. M. Lo, A. T. Cignarella, R. Panizzon, C. Marco, B. Scarlini, V. Patti, C. Bosco, D. Bernardi, EPIC: Multi-perspective annotation of a corpus of irony, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, volume 1, Association for Computational Linguistics, 2023, pp. 13844–13857. doi:10.18653/v1/2023.acl-long.774.

[6] N. R. Bogireddy, S. Suresh, S. Rai, I'm out of breath from laughing! I think? A dataset of COVID-19 humor and its toxic variants, in: Companion Proceedings of the ACM Web Conference 2023, Association for Computing Machinery, New York, NY, 2023, pp. 1004–1013. doi:10.1145/3543873.3587591.

[7] L. Ermakova, T. Miller, A.-G. Bosser, V. M. Palma Preciado, G. Sidorov, A. Jatowt, Overview of JOKER – CLEF-2023 track on automatic wordplay analysis, in: A. Arampatzis, E. Kanoulas, T. Tsikrika, S. Vrochidis, A. Giachanou, D. Li, M. Aliannejadi, M. Vlachos, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction, volume 14163, Springer Nature Switzerland, Cham, 2023, pp. 397–415. doi:10.1007/978-3-031-42448-9_26.

[8] L. Ermakova, A.-G. Bosser, A. Jatowt, T. Miller, The JOKER Corpus: English–French parallel data for multilingual wordplay recognition, in: SIGIR '23: Proceedings of the 46th International

ACM SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery, New York, NY, 2023, pp. 2796–2806. doi:10.1145/3539618.3591885.
[9] D. P. Varadi, A. Bartulović, JOKER 2024 by AB&DPV: From 'LOL' to 'MDR' using AI models to retrieve and translate puns, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[10] S.-H. Wu, Y.-F. Huang, T.-Y. Lau, Humour classification by fine-tuning LLMs: CYUT at CLEF 2024 JOKER Lab subtask humour classification according to genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[11] S. Narayanan, J. J, S. V, CLEF 2024 JOKER Task 3 : Using RoBERTa and BERT-uncased for humour classification according to genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[12] M. Saipranav, J. Sridharan, G. N. G, CLEF 2024 JOKER Task 3: Using BERT and random forest classifier for humor classification according to genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[13] H. Baguian, H. N. Ashley, JOKER Track @ CLEF 2024: The Jokesters' approaches for retrieving, classifying, and translating wordplay, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[14] R. Subramanian, V. Sivaraman, B. B, Humour insights at JOKER 2024 Task 2: Humour classification according to genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[15] V. M. Palma Preciado, C. Palma Preciado, G. Sidorov, NLPalma Joker 2024: Yet, no humor with humorousness – Task 2 humour classification according to genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[16] J. V. Kováčiková, M. Šuppa, Thinking, fast and slow: From the speed of FastText to the depth of retrieval augmented large language models for humour classification, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[17] P. Epron, G. Guibon, M. Couceiro, CLEF 2024 JOKER Task 2: Leveraging large language models for humor genre classification, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[18] R. Elagina, P. Vučić, Convergential approach in machine learning for effective humour analysis and translation, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[19] R. R. Dsilva, N. Bhardwaj, CLEF JOKER 2024 Task 2: Who's laughing now? Humor classification by genre and technique, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[20] R. Mann, T. Mikulandric, CLEF 2024 JOKER Tasks 1–3: Humour identification and classification, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[21] L. Buijs, M. Cazemier, E. Schuurman, J. Kamps, University of Amsterdam at the CLEF 2024 Joker Track, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.
[22] S. S. Balaji, S. N. N.S.R, S. S, Vayam Solve Kurmaha @ CLEF 2024: Task 2: Humor classification according to genre and technique using BERT embeddings and transformers, in: G. Faggioli, N. Ferro, P. Galuscakova, A. García Seco de Herrera (Eds.), Working Notes of CLEF 2024 – Conference and Labs of the Evaluation Forum, 2024.