JOKER TRACK @ CLEF 2024:
AUTOMATIC WORDPLAY ANALYSIS

CONVERGENTIAL APPROACH IN MACHINE LEARNING FOR EFFECTIVE HUMOUR ANALYSIS AND TRANSLATION

Elagina Regina, Vučič Petra

Incorporate team of the Universities of Split and Kiel:
Joker Track

Cognitive Load Theory

Machine Learning Models

Natural Language Processing (NLP)

Cognitive Science

Task 1: Humour-aware Information Retrieval

```
[ ] %cd "/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval"

/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval

[ ] ls

joker_2024_task1_corpus.json       joker_2024_task1_queries_test.json   result_task_1.json
joker_2024_task1_qrels_train.json  joker_2024_task1_queries_train.json

[ ] corpus_data = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_corpus.json")
    corpus_data.head(10)
```

	docid	text
0	1	Good laws have sprung from bad customs.
1	2	The musical score to Topsyturveydom does not s...
2	3	The organic compound primarily responsible for...
3	4	Members of an alliance are called allies.
4	5	Stalk is (verb) To follow or watch someone sec...
5	6	At the 57th Baeksang Arts Awards, it received ...
6	7	The Unchanging is the eighth studio album by A...
7	8	Encephalon is (noun) the brain
8	9	The initial goal is to build a space elevator ...
9	10	In the past, refuse was simply left in piles o...

```
[ ] qrels_train = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_qrels_train.json")
    qrels_train.head(10)
```

	qid	docid	qrel
0	qid_train_0	27260	0

Objective

The primary objective of this task is to develop an effective humour-aware information retrieval system.response to user queries.

Methodology

Data integration, TF-IDF Vectorization, Model training and validation

Approach

The approach involves leveraging a combination of corpus data, query relevance judgments, and training queries to train a model capable of discerning relevant jokes based on queries

Model Setup

- Vectorization
- Machine learning model
- Training data
- Evaluation metrics

2	qid_train_0	51135	1
3	qid_train_0	17068	1
4	qid_train_0	591	1
5	qid_train_1	28237	1
6	qid_train_1	33894	0
7	qid_train_1	42334	0
8	qid_train_1	45620	0
9	qid_train_1	50409	0

```
queries_train = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_queries_train.json")
queries_train.head(10)
```

	qid	query
0	qid_train_0	testament
1	qid_train_1	steps
2	qid_train_10	faculty
3	qid_train_11	death
4	qid_train_2	vein
5	qid_train_3	math
6	qid_train_4	Tom
7	qid_train_5	colors
8	qid_train_6	domestic animal
9	qid_train_7	space

```
import pandas as pd
import json
```

```
with open('/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_qrels_train.json', 'r') as file:
    qrels = json.load(file)
```

```
with open('/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_corpus.json', 'r') as file:
    corpus = json.load(file)
```

```
with open('/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_queries_train.json', 'r') as file:
    train = json.load(file)
```

```
data_qrels = pd.DataFrame(qrels)
data_corpus = pd.DataFrame(corpus)
data_train = pd.DataFrame(train)
```

TF-IDF Vectorizer

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf_vectorizer = TfidfVectorizer()
```

```
#query text and joke text into a single column - TF-IDF Vectorizer
data_merged['text_all'] = data_merged['query'] + " " + data_merged['text']

# Fit and transform the combined text
tfidf_matrix = tfidf_vectorizer.fit_transform(data_merged['text_all'])
```

```
X_train = tfidf_vectorizer.fit_transform(data_merged['text_all'])
y_train = data_merged['qrel']
```

```
from sklearn.linear_model import LogisticRegression

# Logistic Regression model
model = LogisticRegression()

# Trained model
trained_model = model.fit(X_train, y_train)
```

```
model.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
with open('/content/drive/MyDrive/JOKER/JOKER/Task 1 - retrieval/joker_2024_task1_queries_test.json', 'r') as file:
    test_queries = json.load(file)
```

```
data_test_queries = pd.DataFrame(test_queries)
```

```
data_test_queries.head(10)
```

	qid	query
0	qid_test_0	koala
1	qid_test_1	music
2	qid_test_10	children
3	qid_test_11	milk
4	qid_test_12	moonlight
5	qid_test_13	nail
6	qid_test_14	obsession
7	qid_test_15	horse

```
data_test_queries.head(10)
```

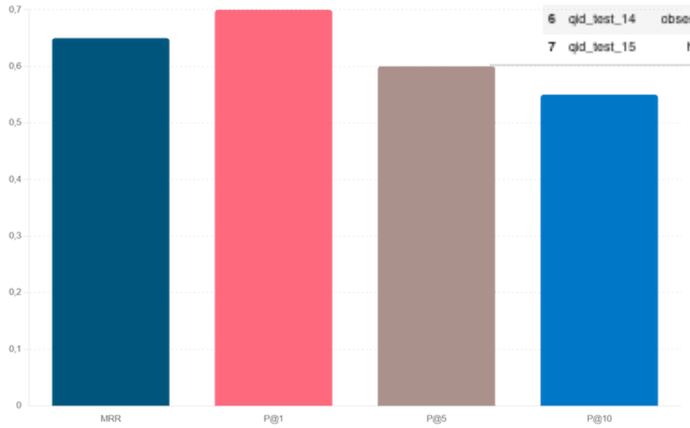
	qid	query
0	qid_test_0	koala
1	qid_test_1	music
2	qid_test_10	children
3	qid_test_11	milk
4	qid_test_12	moonlight
5	qid_test_13	nail
6	qid_test_14	obsession
7	qid_test_15	horse
8	qid_test_16	wild animals
9	qid_test_17	death

```
data_test_queries = data_test_queries.head(5)
```

```
results = []
# Iterate over each test query
for index, test_query in data_test_queries.iterrows():
    query_id = test_query['qid']
    query_text = test_query['query']
    # Calculate relevance for each joke in the corpus with this que
    scores = []
    for _, joke in data_corpus.iterrows():
        if joke['text'] is None:
            continue
        else:
            text_all = query_text + " " + joke["text"]
            vectorized_text = tfidf_vectorizer.transform([text_all])
            relevance_score = model.predict_proba(vectorized_text)[0,
            scores.append({
                'docid': joke['docid'],
                'score': relevance_score
            })
```

```
# Sort jokes by relevance score in descending order
scores.sort(key=lambda x: x['score'], reverse=True)
    # Prepare output JSON format
for rank, score_info in enumerate(scores, start=1):
    results.append({
        'run_id':"team_Petra_and_Regina_task_1_TFIDF",
        'manual':0,
        'rank': rank,
        'score': score_info['score'],
        'docid': score_info['docid'],
        'qid': query_id
    })

with open('result_task_1.json', 'w') as outfile:
    json.dump(results, outfile, indent=4)
```



Metric	Score
MRR	0.65
Precision @ 1	0.70
Precision @ 5	0.60
Precision @ 10	0.55

Task 2: Classification of Humorous Texts

Task 2

```
pwd
```

```
'/content'
```

```
%cd /content/drive/MyDrive/JOKER/JOKER/Task 2 - classification
```

```
/content/drive/MyDrive/JOKER/JOKER/Task 2 - classification
```

```
ls
```

```
joker-2024-task2-classification-test.json          joker-2024-task2-classification-train-qrels.json
joker-2024-task2-classification-train-input.json   result_task_2.json
```

```
classification_test_data = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 2 - classification/joker-2024-task2-classification-test.json")
classification_test_data.head(10)
```

	id	text
0	0	Have you ever felt like your entire life is ju...
1	1	Good day, this is your trashcan speaking.
2	2	My life's purpose is to be a cautionary tale f...
3	3	Yeah, I know. I hate me too.
4	4	"Today is not my day," I mutter to myself ever...
5	5	My teacher called me average. How mean!
6	6	My entire life is a big joke. So, tell why exa...
7	7	I'm actually a very hardworking person. Almost...
8	8	No one can possibly hope to compete with me wh...
9	9	They say money talks. But all mine says is goo...

```
classification_test_data.duplicated().sum()
```

```
0
```

```
classification_train_input_data = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 2 - classification/joker-2024-task2-classification-train-input.json")
classification_train_input_data.head(10)
```

	id	text
0	1162	So in other news I just held the door open for...
1	448	A chimpanzee, a gorilla and a baboon are commu...
2	1280	Amazing how fast this team can go winning from...
3	1216	Thanks for not informing me that my click and...
4	1872	Chemist's work is element-ary.
5	83	I said "hello" to darkness, my old friend, and...
6	65	Having very low expectations is the secret to ...
7	2397	I don't know what possessed me to attend that ...
8	1218	Worked out 2 days in a row. I am quite literal...
9	1661	How do you organize a space party? You planet.

```
classification_train_input_data.duplicated().sum()
```

```
0
```

```
classification_train_qrels_data = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 2 - classification/joker-2024-task2-classification-train-qrels.json")
classification_train_qrels_data.head(10)
```

	id	class
0	1162	SC
1	448	EX
2	1280	SC
3	1216	SC
4	1872	WS
5	83	SD
6	65	SD
7	2397	WS
8	1218	SC
9	1661	AID

```
classification_train_qrels_data.duplicated().sum()
```

```
0
```

```
df_merged_train_data.head(100)
```

	id	text	class
0	2	My life's purpose is to be a cautionary tale f...	SD
1	4	"Today is not my day," I mutter to myself ever...	SD
2	5	My teacher called me average. How mean!	SD
3	6	My entire life is a big joke. So, tell why exa...	SD
4	10	How do I moisturize my face? I use my own tears!	SD
...	...	...	...
95	140	My multitasking abilities – I can only handle ...	SD
96	141	My ability to tell jokes – I tell jokes so unf...	SD
97	142	My karaoke skills – My karaoke performances ar...	SD
98	143	My clumsiness – I can trip over thin air and m...	SD
99	144	I'm quite smart and intelligent. Most of the t...	SD

100 rows × 3 columns

```
import pandas as pd
import json
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
```

```
with open('joker-2024-task2-classification-train-input.json', 'r') as file:
    train_input = json.load(file)
data_train_input = pd.DataFrame(train_input)
```

```
with open('joker-2024-task2-classification-train-qrels.json', 'r') as file:
    train_qrels = json.load(file)
data_train_qrels = pd.DataFrame(train_qrels)
```

```
# Merge on id
dataframe_train = pd.merge(data_train_input, data_train_qrels, on='id')
```

```
# Preprocessing function
def preprocessing_text(text):
    return text.lower()
```

```
dataframe_train['clean_text'] = dataframe_train['text'].apply(preprocessing_text)
```

```
# Encode
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(dataframe_train['class'])
```

```
def preprocessing_text(text):
    return text.lower()
```

```
dataframe_train['clean_text'] = dataframe_train['text'].apply(preprocessing_text)
```

```
# Encode
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(dataframe_train['class'])
```

```
# TF-IDF Vectorization
tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(dataframe_train['clean_text'])
```

```
# Train Logistic Regression model
logistic_regression_model = LogisticRegression(max_iter=1000)
logistic_regression_model.fit(X_train_tfidf, y_train)
```

```
LogisticRegression(max_iter=1000)
```

```
with open('joker-2024-task2-classification-test.json', 'r') as file:
    test_data = json.load(file)
```

```
dataframe_test = pd.DataFrame(test_data)
```

```
# Apply text preprocessing
dataframe_test['clean_text'] = dataframe_test['text'].apply(preprocessing_text)
```

```
# TF-IDF Vectorization for test data
X_test_tfidf = tfidf_vectorizer.transform(dataframe_test['clean_text'])
```

```
test_predictions = logistic_regression_model.predict(X_test_tfidf)
```

```
# Convert back to original names
predicted_classes = label_encoder.inverse_transform(test_predictions)
```

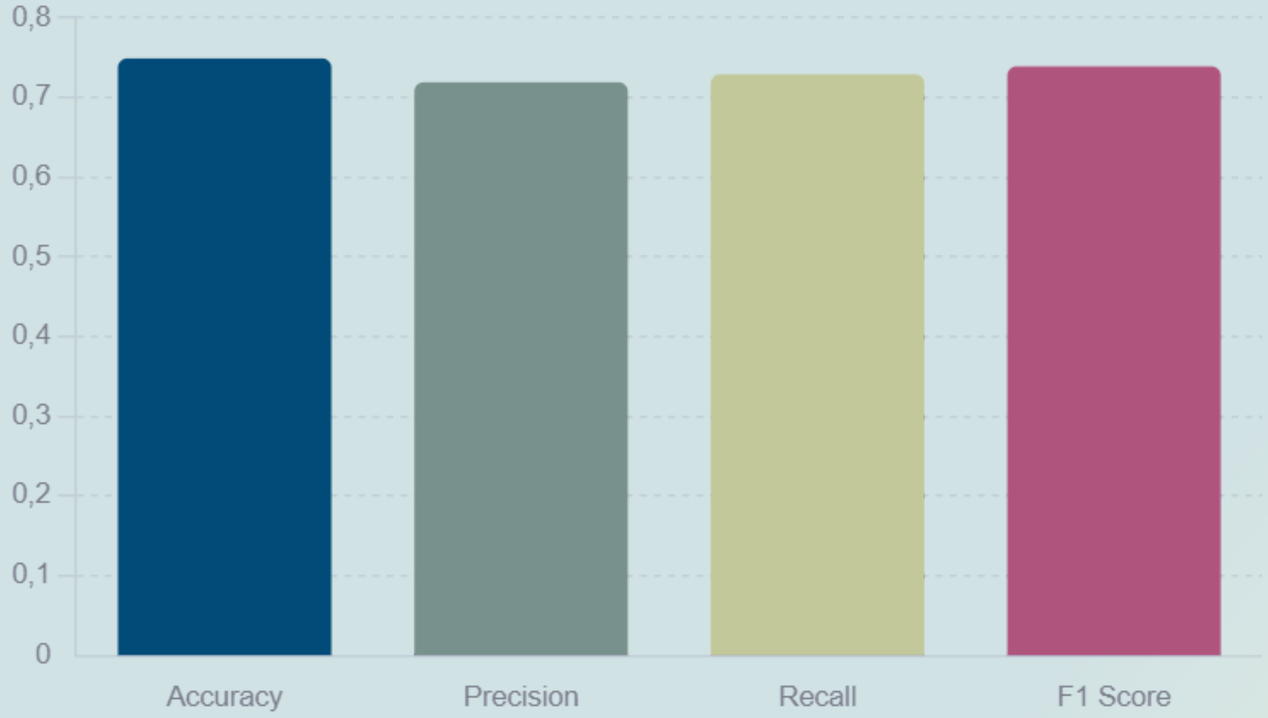
```
results = []
for i, entry in enumerate(test_data):
    output_entry = {
        "run_id": "team1_Petra_and_Regina_task_2_LogisticRegression",
        "manual": 0,
        "id": entry["id"],
        "class": predicted_classes[i]
    }
    results.append(output_entry)
```

```
with open('result_task_2.json', 'w', encoding='utf-8') as outfile:
    json.dump(results, outfile, indent=4)
```

Task 2: Classification of Humorous Texts

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

accuracy = accuracy_score(y_true, y_pred)
precision = precision_score(y_true, y_pred, average='macro')
recall = recall_score(y_true, y_pred, average='macro')
f1 = f1_score(y_true, y_pred, average='macro')



Metric	Score
Accuracy	0.75
Precision @ 1	0.70
Recall	0.73
F1 Score	0.74

Incorporate team of the Universities of Split and Kiel:
Joker Track

Task 3: Translation of Puns from English to French

Task 3: Translation of puns from English to French

```
pwd
```

```
'/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train'
```

```
%cd "/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train"
```

```
/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train
```

```
ls
```

```
joker_task3_2024_test.json                joker_translation_EN-FR_train_qrels.json
joker_translation_EN-FR_train_input.json  joker_translation_EN-FR_train_qrels.tsv
joker_translation_EN-FR_train_input.tsv
```

```
translation_EN_FR_train_input = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train/joker_translation_EN-FR_train_input.json")
translation_EN_FR_train_input.head(5)
```

	id_en	text_en
0	en_1007	Save the whales, spouted Tom.
1	en_1015	If a town's people have low IQs is the populat...
2	en_102	A skier retired because he was going downhill.
3	en_103	My wife uses a kitchen implement to shred garl...
4	en_1031	Staying at the trendy, new hotel was the inn t...

```
translation_EN_FR_train_input.duplicated().sum()
```

```
0
```

```
task3_2024_test = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train/joker_task3_2024_test.json")
task3_2024_test.head(5)
```

	id_en	text_en
0	en_0	To get 20/20 in a marathon you really need to ...
1	en_1	The maths teacher always arrives wearing an od...
2	en_2	What is the worst thing about being a director...
3	en_3	There was a guy who got fired from the orange ...
4	en_4	Old presidents never die, they're just out of ...

```
0
```

```
translation_EN_FR_train_qrels = pd.read_json("/content/drive/MyDrive/JOKER/JOKER/Task 3 - translation/EN-FR-train/joker_translation_EN-FR_train_qrels.json")
translation_EN_FR_train_qrels.head(5)
```

	id_en	text_fr
0	en_1007	"Il faut sauver les baleines", jeta Tom avant ...
1	en_1007	"Il faut sauver les baleines", interjeta Tom.
2	en_1007	Moi je sauve les baleines, Tom s'en venta.
3	en_1007	Louis évent-a le projet de sauvetage des balei...
4	en_1007	"Sauvez les baleines", proclama Tom à tout évent.

```
translation_EN_FR_train_qrels.duplicated().sum()
```

```
0
```

```
dataframe_merged_translate = pd.merge(translation_EN_FR_train_input, translation_EN_FR_train_qrels, on='id_en')
```

```
dataframe_merged_translate.tail(5)
```

	id_en	text_en	text_fr
5833	en_965	Sea captains have a lot of latitude.	Les capitaines de navire ont beaucoup de latit...
5834	en_968	Last night, I kept dreaming that I had written...	J'ai rêvé que j'étais l'auteur du Seigneur des...
5835	en_99	How does a card player party? They shuffle.	Échauffourée au club de bridge : deux personne...
5836	en_99	How does a card player party? They shuffle.	Ils font quoi, les joueurs de cartes, au son d...
5837	en_99	How does a card player party? They shuffle.	Comment se passe une soirée de joueurs de cart...

```
import pandas as pd
import json
```

```
with open('joker_translation_EN-FR_train_input.json', 'r') as file:
    train_input = json.load(file)
dataframe_train_input = pd.DataFrame(train_input)
```

```
with open('joker_translation_EN-FR_train_qrels.json', 'r') as file:
    train_qrels = json.load(file)
dataframe_train_qrels = pd.DataFrame(train_qrels)
```

```
# Training data with labels on id
dataframe_train = pd.merge(dataframe_train_input, dataframe_train_qrels, on='id_en')
```

```
dataframe_train.tail(5)
```

	id_en	text_en	text_fr
5833	en_965	Sea captains have a lot of latitude.	Les capitaines de navire ont beaucoup de latit...
5834	en_968	Last night, I kept dreaming that I had written...	J'ai rêvé que j'étais l'auteur du Seigneur des...
5835	en_99	How does a card player party? They shuffle.	Échauffourée au club de bridge : deux personne...
5836	en_99	How does a card player party? They shuffle.	Ils font quoi, les joueurs de cartes, au son d...
5837	en_99	How does a card player party? They shuffle.	Comment se passe une soirée de joueurs de cart...

```
!pip install -U easynmt
```

```
Requirement already satisfied: easynmt in /usr/local/lib/python3.10/dist-packages (2.0.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from easynmt) (4.66.4)
Requirement already satisfied: transformers<5,>=4.4 in /usr/local/lib/python3.10/dist-packages (from easynmt) (4.40.2)
Requirement already satisfied: torch>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from easynmt) (2.2.1+cu121)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from easynmt) (1.25.2)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from easynmt) (3.8.1)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.10/dist-packages (from easynmt) (0.1.99)
Requirement already satisfied: fasttext in /usr/local/lib/python3.10/dist-packages (from easynmt) (0.9.2)
Requirement already satisfied: protobuf in /usr/local/lib/python3.10/dist-packages (from easynmt) (3.20.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (3.14.0)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (4.11.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.19.3 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (2.19.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (12.1.105)
Requirement already satisfied: triton==2.2.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->easynmt) (2.2.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in /usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-cu12==11.4.5.107->torch>=1.6.0->easynmt) (12.4.127)
Requirement already satisfied: huggingface-hub<1.0,>=0.19.3 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (0.20.3)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (24.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (2023.12.25)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (2.31.0)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (0.19.1)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers<5,>=4.4->easynmt) (0.4.3)
Requirement already satisfied: pybind11>=2.2 in /usr/local/lib/python3.10/dist-packages (from fasttext->easynmt) (2.12.0)
Requirement already satisfied: setuptools>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from fasttext->easynmt) (67.7.2)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->easynmt) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->easynmt) (1.4.2)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.6.0->easynmt) (2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers<5,>=4.4->easynmt) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers<5,>=4.4->easynmt) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local
Requirement already satisfied: certifi>=2017.4.17 in /usr/local
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/p
```

```
from easynmt import EasyNMT
model = EasyNMT('opus-mt')
```

```
results = []
for text_en in dataframe_train['text_en'][:2]:  # Accessing the first 2 elements
    result = model.translate(text_en, target_lang='fr')
    results.append(result)
```

```
print(result)
```

```
Sauvez les baleines, a soufflé Tom.
```

```
with open('joker_task3_2024_test.json', 'r') as file:
    test_data = json.load(file)
```

```
dataframe_test_data = pd.DataFrame(test_data)
dataframe_test_data.head(10)
```

```
print(type(dataframe_test_data))  # Check the type of dataframe_test_data

# Print the first few rows to inspect the structure
print(dataframe_test_data.head())
```

```
<class 'pandas.core.frame.DataFrame'>
  id_en                                            text_en
0  en_0  To get 20/20 in a marathon you really need to ...
1  en_1  The maths teacher always arrives wearing an od...
2  en_2  What is the worst thing about being a director...
3  en_3  There was a guy who got fired from the orange ...
4  en_4  Old presidents never die, they're just out of ...
```

```
dataframe_test_data = dataframe_test_data.head()
```

```
results = []
# Translate jokes
for index, row in dataframe_test_data.iterrows():  # Iterate over rows
    translation = model.translate(dataframe_test_data['text_en'], source_lang='en', target_lang='fr')  # Translate individual text
```

Task 3: Translation of Puns from English to French

```
result = model.translate(text_en, target_lang='fr')
```

For example, the pun

“Save the whales, Tom blubbered.”

was translated as:

"Sauvez les baleines, a soufflé Tom."

THANK YOU

Incorporate team of the Universities of Split and Kiel:
Joker Track