Here is a clean, practical list of **which hyperparameters Optuna can and should tune** when you're doing **multiclass classification using RoBERTa** (I assume "Roberts" was meant to be *RoBERTa*).
This applies whether you're using **Hugging Face Transformers**, **PyTorch Lightning**, or a custom training loop.

✅ Hyperparameters You Can Optimize with Optuna (RoBERTa Multiclass)

Below is the complete checklist—grouped for clarity.

🔷 1. Learning Rate & Schedulers

These usually give the biggest improvement.

Parameter	Typical Range	Notes
`learning_rate`	**1e-6 – 5e-5**	Most important hyperparameter
`weight_decay`	**0 – 0.3**	Prevents overfitting
`warmup_ratio`	**0.0 – 0.2**	Smooth start helps stable training
`adam_epsilon`	**1e-9 – 1e-7**	Usually stable around 1e-8
`lr_scheduler_type`	Choose: `linear`, `cosine`, `cosine_with_restarts`, `polynomial`	Use Optuna's `trial.suggest_categorical`

🔷 2. Batch & Training Steps

These must match your GPU memory.

Parameter	Range	Notes

`batch_size`	**8, 16, 32**	If OOM → reduce
`gradient_accumulation_steps`	**1–8**	Helps simulate larger batch size
`num_train_epochs`	**2–6**	RoBERTa often needs 3–4

🔷 3. RoBERTa Model-Specific Params

Even fine-tuning can benefit from these (if using HF Trainer).

Parameter	Range	Notes
`max_seq_length`	**64–512**	Huge effect on quality + speed
`attention_dropout`	**0.0–0.3**	Prevents attention overfitting
`hidden_dropout`	**0.0–0.3**	Regularization
`classifier_dropout`	**0.0–0.3**	Extra dropout on classification head
`layerwise_lr_decay`	**0.8–1.0**	Lower LR for early layers (optional)

🔷 4. Optimization Algorithms

You can also tune:

Parameter	Range
`optimizer_type`	AdamW, Adafactor
`beta1`, `beta2`	(0.7–0.95), (0.95–0.999)

Usually AdamW works best.

🔷 5. Regularization & Early Stopping

Helps avoid overfitting on multi-class problems:

Parameter	Range	Notes
`label_smoothing_factor`	**0.0 – 0.2**	VERY useful for multi-class
`early_stopping_patience`	**1–5**	Controlled by Trainer callback

🧪 Complete Optuna Search Space Template (Copy-Paste Ready)

```
def optuna_hp_space(trial):
    return {
        "learning_rate": trial.suggest_float("learning_rate", 1e-6, 5e-5, log=True),
        "weight_decay": trial.suggest_float("weight_decay", 0.0, 0.3),
        "warmup_ratio": trial.suggest_float("warmup_ratio", 0.0, 0.2),
        "adam_epsilon": trial.suggest_float("adam_epsilon", 1e-9, 1e-7, log=True),

        "num_train_epochs": trial.suggest_int("num_train_epochs", 2, 6),
        "per_device_train_batch_size": trial.suggest_categorical("batch_size", [8, 16, 32]),
        "gradient_accumulation_steps": trial.suggest_int("grad_accum", 1, 8),

        "lr_scheduler_type": trial.suggest_categorical(
            "lr_scheduler_type",
            ["linear", "cosine", "cosine_with_restarts", "polynomial"]
        ),
```

```
        "max_seq_length": trial.suggest_categorical("max_seq_length",
[64, 128, 256, 512]),
        "hidden_dropout_prob":
trial.suggest_float("hidden_dropout_prob", 0.0, 0.3),
        "attention_probs_dropout_prob":
trial.suggest_float("attention_dropout", 0.0, 0.3),
        "classifier_dropout":
trial.suggest_float("classifier_dropout", 0.0, 0.3),

        "label_smoothing_factor":
trial.suggest_float("label_smoothing", 0.0, 0.2),

        "layerwise_lr_decay":
trial.suggest_float("layerwise_lr_decay", 0.8, 1.0)
    }
```

DH write it in a style which is fit for college project writing in a group

Lets use SpaCy to clean the data, tokenize but dont remove stop words ,dont remove puntuations, space i.e full normalizationnot doing lemmatization as we want the LLM to do its job. . We are removing any HTML tags and URLs as that is challanging for the Model to run. We are also removing and word repeated by mistake. For HTML and URL cleaning we are using beautiful soup . We ar eusing Spacy for pipeline as that we found very easy to setup. There are other options that we considered like

Show more

Restructured informal technical content into formal academic prose.

markdown

```
## Data Preprocessing Pipeline

### Preprocessing Approach

For our data preprocessing pipeline, we have implemented SpaCy to perform tokenizati

### Data Cleaning Strategy

Our cleaning methodology focuses on removing elements that pose challenges for model
- HTML tags
- URLs
- Accidentally repeated words

For HTML and URL removal, we have selected BeautifulSoup due to its robust parsing c

### Pipeline Architecture

After evaluating several pipeline frameworks, we chose to implement a hybrid approac
- **Apache Airflow**: A DAG-based workflow management platform that offers sophistic
- **Azure Data Factory**: Cloud-based data integration service
- **Regex-based solutions**: While powerful, this approach would require extensive p

Given our development environment in Google Colab, we determined that SpaCy provides
```

Retry

DH rewrite this

Our overall strategy for the purpose of tuning is :
* get a stratified dataset of smaller size - we will do 20% as otherwise certain classes have very low no of records
* Try to find which optimization algorithm , with what learning rate and other parameters gives the best results for what batch size (The algorithms considered are "adam", "adamw", "sgd", "rmsprop", "adagrad". Other options are not using - Adamax , Nadam , Adadelta , Ftrl . This all was arrived at based on research on internet as to which one are most suitable for multclass classification where the boundaries are not

Show more

Restructured hyperparameter tuning strategy into professional academic framework.

markdown

```
## Hyperparameter Optimization Strategy

Our comprehensive tuning approach encompasses multiple dimensions of model configura

### Data Sampling and Class Balance

We will utilize a **stratified 20% sample** of the full dataset for hyperparameter t

### Optimization Algorithm Selection

Based on literature review and empirical research on multi-class classification with

**Selected Algorithms:**
- Adam
- AdamW
- SGD (Stochastic Gradient Descent)
- RMSprop
- Adagrad

**Excluded Algorithms:** Adamax, Nadam, Adadelta, and Ftrl were not included based o

### Model Architecture Configurations

Our experimentation will cover three transformer-based models:
- **BERT** (Bidirectional Encoder Representations from Transformers)
- **RoBERTa** (Robustly Optimized BERT Pretraining Approach)
- **DistilBERT** (Distilled version of BERT)
```

Session limit reached • resets 4:00 PM

Keep working

DH reqrite this

In milestone 2 we used the max token lenght of text of 27 and we had used lemmatization, removing stopwords and punctuations. Since the performance of the model was not that good , so we decided to use 99 percentile of the cleaned text for the purpose of texting(ie max length of the text to be used for train, validate and test). This resulted to us getting MAX_LEN of 75 instead of 27. THis is also due to the fact we had removed stopwords, punctualtions besides using lemmatization

Restructured technical documentation for academic presentation clarity.

markdown

```
### Evolution of Text Length Parameters

In Milestone 2, our initial preprocessing pipeline included aggressive text normaliz

For the current iteration, we have revised our approach based on the following ratio

**Revised Preprocessing Strategy:**
- Eliminated lemmatization to preserve word forms and contextual nuances
- Retained stopwords and punctuation to maintain sentence structure and meaning
- Calculated maximum sequence length using the 99th percentile of the cleaned text c

**Impact on Model Parameters:**
This modification has increased our MAX_LEN parameter from 27 to 75 tokens. This sig
```

DH rewrite this

From above we can infer there are 41 classes** and inorder to get a better trained model we need about 20% of data (as then the smallest class will have between 150 and 200 records) with some rows expected to drop due to data quality . With that we will have minimum data needed per class

Elevated markdown text to formal academic writing style.

markdown

```
### Data Sampling Strategy

Our analysis reveals that the dataset contains **41 distinct categories**. To ensure

- The smallest category will contain approximately 150-200 records after sampling
- We anticipate some data loss during quality filtering processes
- This sample size meets the minimum threshold required for each class to support re

By maintaining this 20% sampling ratio, we ensure that even our least-represented ca
```

DH rewrite this
Overall trainign strategy
* get the best hyperparameters fom optuna
* reconstruct the model
* 1st Round training- train the model with 20% stratified data
* Understand classification issues using F1 score by class and the support(i.e. number of records) for that category
* Deal withose classes (strategy adopted- Merging labels whciha re very similar content based on the classification report and TF-IDF cosine similarity Matrix

Show more

Restructured iterative model training pipeline into academic format.

markdown

```
## Overall Training Strategy

Our iterative training approach consists of multiple phases designed to progressivel

### Phase 1: Initial Hyperparameter Optimization and Training

**1.1 Hyperparameter Search**
   - Execute Optuna optimization framework to identify optimal hyperparameters acros
   - Search space includes: optimizer type, learning rates, batch sizes, and fine-tu

**1.2 First Round Training (20% Stratified Sample)**
   - Train models using the best hyperparameters identified by Optuna
   - Utilize stratified 20% sample to ensure class representation while maintaining
   - Evaluate initial model performance using validation metrics

### Phase 2: Performance Analysis and Label Strategy Development

**2.1 Classification Issue Diagnosis**
   - Analyze per-class F1 scores to identify poorly performing categories
   - Examine class support (number of training samples) to identify data scarcity is
   - Generate classification reports and confusion matrices to understand misclassif

**2.2 Label Merging Strategy**
   - Identify semantically similar categories exhibiting high confusion rates
   - Apply TF-IDF cosine similarity analysis to quantify content overlap between cat
   - Merge categories based on both cla ification confusion patterns and content si
   - Document merged category mappings  reproducibility
```

Session limit reached • resets 4:00 PM Keep working

DH write it in a style which is fit for college project writing in a group

Lets use SpaCy to clean the data, tokenize but dont remove stop words ,dont remove puntuations, space i.e full normalizationnot doing lemmatization as we want the LLM to do its job. . We are removing any HTML tags and URLs as that is challanging for the Model to run. We are also removing and word repeated by mistake. For HTML and URL cleaning we are using beautiful soup . We ar eusing Spacy for pipeline as that we found very easy to setup. There are other options that we considered like

Show more

Restructured informal technical content into formal academic prose.

markdown

```
## Data Preprocessing Pipeline

### Preprocessing Approach

For our data preprocessing pipeline, we have implemented SpaCy to perform tokenizati

### Data Cleaning Strategy

Our cleaning methodology focuses on removing elements that pose challenges for model
- HTML tags
- URLs
- Accidentally repeated words

For HTML and URL removal, we have selected BeautifulSoup due to its robust parsing c

### Pipeline Architecture

After evaluating several pipeline frameworks, we chose to implement a hybrid approac
- **Apache Airflow**: A DAG-based workflow management platform that offers sophistic
- **Azure Data Factory**: Cloud-based data integration service
- **Regex-based solutions**: While powerful, this approach would require extensive p

Given our development environment in Google Colab, we determined that SpaCy provides
```

DH use this code to shorotn comments and remove typo

Identified typos and streamlined verbose comments systematically.

python

```python
# Experiment loop with Optuna hyperparameter optimization

from sklearn.metrics import classification_report, confusion_matrix, f1_score
import keras.backend as K
import gc


def run_experiment_loop_with_optuna(
        X_tr, X_va, X_te, y_tr, y_va, y_te,
        current_subset_frac,
        model_configs_list,
        max_epochs=10,
        num_classes_global=num_classes,
        label_names_global=label_names,
        class_weights_global=class_weights,
        DS_Merged=False,
        OPTUNA_TRIALS=10,
        ):
    local_results = {}
    local_subset_stats = []
    current_study = None  # Stores the last created study object
    SHORT_EPOCH_OPTUNA_SEARCH = 5  # Short Optuna search to suggest optimized hyperp

    print(f"\n{'-'*80}")
    print(f"OPTUNA RUNNING FOR DATASET SUBSET FRACTION: {current_subset_frac}")
    print(f"{'-'*80}\n")

    print(f"Data - Train: {len(X_tr)}, Val: {len(X_va)}, Test: {len(X_te)}")

    merged_categories = "Few classes Merged" if DS_Merged else "No Merged Class"

    # Loop through model configurations
    for cfg in model_configs_list:
        model_name = cfg.get("name")
        strategy = cfg.get("strategy", "freeze")
        n_partial_layers = cfg.get("unf e_layers", None)
        cfg_batch_value = cfg.get("batch , BATCHS)
```

Session limit reached • resets 4:00 PM Keep working