Package: conflibertR (via r-universe)

July 11, 2026

Title Inference and Fine-Tuning with 'ConfliBERT' Conflict Text Models

Version 0.5.3

Description An interface to 'ConfliBERT', a pretrained language model for analyzing text about conflict and political violence (Hu et al. (2022) <doi:10.18653/v1/2022.naacl-main.400>). Provides functions for named entity recognition, binary and multilabel classification, and question answering, plus tools to fine-tune custom classifiers, compare several base model architectures, and run an interactive active-learning loop for efficiently labeling new data. Models are downloaded from 'Hugging Face' and run through the 'transformers' library for 'Python' via the 'reticulate' package.

License MIT + file LICENSE

URL https://github.com/shreyasmeher/conflibertR

BugReports https://github.com/shreyasmeher/conflibertR/issues

Encoding UTF-8

Language en

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

SystemRequirements Python (>= 3.8); the Python modules 'torch', 'transformers' (>= 4.40), 'accelerate', 'peft', 'scikit-learn' and 'numpy', installable via conflibert_install()

Imports cli, reticulate (>= 1.34), tibble

Suggests ggplot2, knitr, miniUI, rmarkdown, rstudioapi, shiny, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/pak/sysreqs libpng-dev python3

Repository https://shreyasmeher.r-universe.dev

Date/Publication 2026-07-03 06:08:33 UTC

RemoteUrl https://github.com/shreyasmeher/conflibertr

RemoteRef HEAD

RemoteSha 8bc8707d913e6d8112a382e4ba1c078718f2aa6b

Contents

autoplot.conflibert_al_session

Plot Active Learning Progress with ggplot2

Description

ggplot2 version of `plot.conflibert_al_session`: the learning curve and the query-uncertainty trend across rounds.

Usage

```
## S3 method for class 'conflibert_al_session'
autoplot(object, which = c("all", "metrics", "uncertainty"), ...)
```

Arguments

`object`	A `conflibert_al_session`.
`which`	`"all"` (default), `"metrics"`, or `"uncertainty"`.
`...`	Ignored.

Value

A ggplot object.

autoplot.conflibert_classify

Plot Classification Results with ggplot2

Description

Aggregated view of a `conflibert_classify` result: how many texts landed in each class, annotated with the average confidence.

Usage

```
## S3 method for class 'conflibert_classify'
autoplot(object, ...)
```

Arguments

`object`	A result from `conflibert_classify`.
`...`	Ignored.

Value

A ggplot object.

autoplot.conflibert_comparison

Plot a Model Comparison with ggplot2

Description

Dot chart of every metric for every model in a `conflibert_compare` result, models ordered by their primary metric.

Usage

```
## S3 method for class 'conflibert_comparison'
autoplot(object, ...)
```

Arguments

`object`	A result from `conflibert_compare`.
`...`	Ignored.

Value

A ggplot object.

autoplot.conflibert_finetune

Plot a Fine-tuning Confusion Matrix with ggplot2

Description

Test-set confusion matrix heatmap for a `conflibert_finetune` result.

Usage

```
## S3 method for class 'conflibert_finetune'
autoplot(object, ...)
```

Arguments

`object`	A `conflibert_finetune` result.
`...`	Ignored.

Value

A ggplot object.

autoplot.conflibert_multilabel

Plot Multilabel Results with ggplot2

Description

Plots a `conflibert_multilabel` result. With several texts it aggregates: the share of texts flagged for each event category. With a single text it shows that text's category probabilities against the 0.5 decision threshold.

Usage

```
## S3 method for class 'conflibert_multilabel'
autoplot(object, ...)
```

Arguments

`object`	A result from `conflibert_multilabel`.
`...`	Ignored.

Value

A ggplot object.

autoplot.conflibert_ner

Plot NER Results with ggplot2

Description

Aggregated view of a `conflibert_ner` result: `type = "types"` (default) counts entities per entity type; `type = "entities"` shows the most frequent individual entities, colored by type.

Usage

```
## S3 method for class 'conflibert_ner'
autoplot(object, type = c("types", "entities"), top_n = 12, ...)
```

Arguments

`object`	A result from `conflibert_ner`.
`type`	`"types"` or `"entities"`.
`top_n`	How many entities to show when `type = "entities"`. Default: 12.
`...`	Ignored.

Value

A ggplot object.

`conflibert_active_label` *Label the Current Query Interactively*

Description

Opens a small Shiny gadget that shows each text in `session$query` alongside radio buttons for each class. Click Done to submit; the labels are returned as an integer vector ready to pass to `conflibert_active_next`.

Usage

```
conflibert_active_label(session, classes = NULL)
```

Arguments

`session`	A `conflibert_al_session` object with a non-empty `query`.
`classes`	Optional named integer vector mapping display names to class values, e.g. `c("non-conflict" = 0, "conflict" = 1)`. If omitted, classes are inferred from the session.

Details

Requires the **shiny** and **miniUI** packages. In RStudio the gadget opens as a modal dialog; elsewhere it opens in your browser.

Value

An integer vector of labels (one per query row), or `NULL` if the user cancels.

Examples

```
data <- conflibert_example("active")
session <- conflibert_active_start(
  seed = data$seed, pool = data$pool, query_size = 5, epochs = 1
)
labels  <- conflibert_active_label(session)
session <- conflibert_active_next(session, labels)
```

conflibert_active_next

Submit Labels and Query the Next Batch

Description

Takes a session from `conflibert_active_start` (or a previous call to this function), incorporates your labels for the current query, retrains the model on the full labeled set, and selects the next uncertain batch from the remaining pool.

Usage

```
conflibert_active_next(session, labels)
```

Arguments

`session`	A `conflibert_al_session` object.
`labels`	Integer (or coercible) vector of labels for `session$query`, in the same order. Length must match `nrow(session$query)`.

Value

An updated `conflibert_al_session`. When the pool is exhausted, `session$done` is `TRUE` and `session$query` is empty.

Examples

```
data <- conflibert_example("active")
session <- conflibert_active_start(
  seed = data$seed, pool = data$pool, query_size = 5, epochs = 1
)
# label the queried texts (here using the bundled oracle labels)
labels  <- unname(data$pool_labels[session$query$text])
session <- conflibert_active_next(session, labels)
session$metrics
```

conflibert_active_save

Save the Active Learning Model

Description

Write the current session's model and tokenizer to disk. The result is a standard HuggingFace checkpoint that can be reloaded with any `transformers` tool.

Usage

```
conflibert_active_save(session, dir)
```

Arguments

session	A conflibert_al_session object.
dir	Directory to write the model into. Created if it does not exist.

Value

The directory path, invisibly.

Examples

```
data <- conflibert_example("active")
session <- conflibert_active_start(
  seed = data$seed, pool = data$pool, query_size = 5, epochs = 1
)
dir <- file.path(tempdir(), "my_al_model")
conflibert_active_save(session, dir)
```

---

conflibert_active_start

Start an Active Learning Session

Description

Train a classifier on a small labeled seed and select the most uncertain samples from an unlabeled pool for you to label. The returned session object tracks the labeled set, the pool, the current query, and metrics across rounds.

Usage

```
conflibert_active_start(
  seed,
  pool,
  dev = NULL,
  model = "ConfliBERT",
  task = c("binary", "multiclass"),
  strategy = c("entropy", "margin", "least_confidence"),
  diverse = FALSE,
  diversity_candidates = NULL,
  query_size = 10,
  epochs = 3,
  batch_size = 8,
  lr = 2e-05,
```

```
  max_seq_len = 512,
  use_lora = FALSE,
  lora_rank = 8,
  lora_alpha = 16,
  random_seed = 42
)
```

Arguments

seed	Labeled starter set. A data.frame with `text` and `label` columns.
pool	Unlabeled pool. Either a character vector of texts, or a data.frame with a `text` column (other columns are ignored).
dev	Optional validation set with `text` and `label` columns. When provided, metrics are recorded each round.
model	Base model. See `conflibert_models`. Default: `"ConfliBERT"`.
task	`"binary"` or `"multiclass"`. Default: `"binary"`.
strategy	Uncertainty strategy: `"entropy"` (default), `"margin"`, or `"least_confidence"`.
diverse	If `TRUE`, cluster the top-scoring candidates in embedding space and pick the highest-scoring sample from each cluster. Prevents near-duplicates from dominating a batch. Default: `FALSE`.
diversity_candidates	How many top-scoring candidates to cluster when `diverse = TRUE`. Default: `3 * query_size`.
query_size	Samples queried per round. Default: 10.
epochs	Training epochs per round. Default: 3.
batch_size	Training batch size. Default: 8.
lr	Learning rate. Default: 2e-5.
max_seq_len	Max token sequence length. Default: 512.
use_lora	If `TRUE`, train each round with a LoRA adapter (parameter-efficient). Default: `FALSE`.
lora_rank	LoRA rank. Default: 8.
lora_alpha	LoRA alpha. Default: 16.
random_seed	Random seed for reproducibility (named to avoid a clash with the labeled `seed` set above). Seeds each round's model training, so re-running the whole session on the same hardware and package versions reproduces the same queries and metrics. (Diverse query selection with `diverse = TRUE` is deterministic and needs no seed.) It is stored in the session and reused by `conflibert_active_next`. Default: 42.

Details

The typical workflow is:

1. Call `conflibert_active_start()` with your seed and pool.

2. Inspect `session$query` and assign labels.
3. Pass the labels to `conflibert_active_next` to retrain and query the next batch.
4. Repeat until the pool is exhausted or metrics plateau.
5. Persist with `conflibert_active_save`.

Value

An object of class `"conflibert_al_session"`: a list with `query` (tibble of texts to label), `metrics` (tibble of metrics across rounds), `round`, `labeled_n`, `pool_n`, `done`, and internal state.

Examples

```
data <- conflibert_example("active")
session <- conflibert_active_start(
  seed = data$seed, pool = data$pool, query_size = 5, epochs = 1
)
session
session$query
```

`conflibert_available` *Is the ConfliBERT Python Backend Available?*

Description

Returns `TRUE` when the `"conflibert"` Python environment exists and the core Python modules ('torch' and 'transformers') can be imported. It never installs anything; environment discovery uses filesystem checks only (it does not run the conda binary), and when no environment is found it returns `FALSE` without initializing Python, so it is cheap and safe to call on any system. It is used to guard the package's examples on machines without the backend (such as CRAN's check machines), and you can use it the same way in scripts that should degrade gracefully.

Usage

```
conflibert_available()
```

Details

The detection result is cached for the R session (if you have just run `conflibert_install`, restart R as its instructions say). Set the environment variable `CONFLIBERTR_AVAILABLE` to `"true"` or `"false"` to override the detection, e.g. to skip the backend-dependent examples during `R CMD check` on a machine that has the backend installed.

Value

`TRUE` or `FALSE`.

Examples

```
conflibert_available()
```

---

conflibert_benchmark *Benchmark the Pretrained Classifier*

Description

Evaluate the pretrained ConfliBERT binary classifier against labeled data and compute accuracy, precision, recall, and F1.

Usage

```
conflibert_benchmark(texts, labels)
```

Arguments

texts	Character vector of texts.
labels	Integer vector of true labels (0 or 1).

Value

A tibble with one row and columns: accuracy, precision, recall, f1, n.

Examples

```
conflibert_benchmark(
  texts  = c("A bomb exploded.", "The weather was nice."),
  labels = c(1L, 0L)
)
```

---

conflibert_classify *Binary Classification*

Description

Classify text as conflict-related (Positive) or not (Negative) using the pretrained ConfliBERT binary classifier. Inference is batched, so long vectors are fast; a progress bar appears for large inputs.

Usage

```
conflibert_classify(text)
```

Arguments

`text`	A character vector of one or more texts.

Value

A tibble (with a themed print method) with columns:

text The input text.

label "Positive" or "Negative".

class Integer (1 = positive, 0 = negative).

confidence Probability of the predicted class.

prob_negative Probability of the negative class.

prob_positive Probability of the positive class.

Examples

```
conflibert_classify("A bomb exploded in the market.")

conflibert_classify(c(
  "Government troops clashed with rebels.",
  "The weather was sunny and warm."
))
```

`conflibert_compare` *Compare Multiple Models*

Description

Fine-tune several base models on the same dataset and return a comparison table of test-set metrics. Useful for selecting the best architecture for your data.

Usage

```
conflibert_compare(
  train,
  dev,
  test,
  models = c("ConfliBERT", "BERT Base Uncased"),
  task = "binary",
  epochs = 3,
  batch_size = 8,
  lr = 2e-05,
  use_lora = FALSE,
  lora_rank = 8,
  lora_alpha = 16,
  seed = 42
)
```

Arguments

train	A data.frame with text and label columns (training set).
dev	A data.frame with text and label columns (validation set).
test	A data.frame with text and label columns (test set).
models	Character vector of model names to compare. See conflibert_models for available names.
task	"binary" or "multiclass". Default: "binary".
epochs	Number of training epochs. Default: 3.
batch_size	Training batch size. Default: 8.
lr	Learning rate. Default: 2e-5.
use_lora	If TRUE, fine-tune with a LoRA adapter (parameter-efficient; cuts GPU memory roughly 5x). The adapter is merged into the base model before saving, so loading is the same as for full fine-tuning. Default: FALSE.
lora_rank	LoRA rank. Default: 8.
lora_alpha	LoRA alpha. Default: 16.
seed	Random seed for reproducibility. Seeds the classifier-head initialization, data shuffling, and dropout so that two runs with the same seed on the same hardware and package versions give identical results. Change it to study run-to-run variability. Default: 42.

Value

A tibble with one row per model and columns for each metric plus runtime. It has a themed print() method that ranks models, and a plot() method comparing metrics.

Examples

```
data <- conflibert_example("binary")
comparison <- conflibert_compare(
  train  = data$train,
  dev    = data$dev,
  test   = data$test,
  models = c("ConfliBERT", "BERT Base Uncased"),
  task   = "binary",
  epochs = 1
)
comparison
```

`conflibert_example` *Load Example Dataset*

Description

Load one of the bundled example datasets for testing fine-tuning and model comparison. Returns a list with train, dev, and test data frames ready to pass to `conflibert_finetune` or `conflibert_compare`.

Usage

```
conflibert_example(name = c("binary", "multiclass", "active"))
```

Arguments

`name`	One of: `"binary"` conflict vs non-conflict, 2 classes. Returns `$train`, `$dev`, `$test`. `"multiclass"` 4 event types: Diplomacy, Armed Conflict, Protest, Humanitarian. Returns `$train`, `$dev`, `$test`. `"active"` small labeled seed and an unlabeled pool for demoing `conflibert_active_start`. Returns `$seed`, `$pool`, `$dev`, and `$pool_labels`, a named integer vector mapping each pool text to its true label (for simulation / automated testing only; do not use these in a real workflow).

Value

A named list of data frames (and a character vector for the active-learning pool). See `name` for the shape per dataset.

Examples

```
# Loading the bundled data is pure R and needs no Python backend:
data <- conflibert_example("binary")
data$train


# Fine-tune with example data (needs the Python backend)
data <- conflibert_example("binary")
result <- conflibert_finetune(
  train = data$train, dev = data$dev, test = data$test,
  model = "ConfliBERT", task = "binary", epochs = 1
)

# Compare models with example data
comparison <- conflibert_compare(
  train = data$train, dev = data$dev, test = data$test,
  models = c("ConfliBERT", "BERT Base Uncased"),
  task = "binary", epochs = 1
)
```

conflibert_finetune *Fine-tune a Classification Model*

Description

Train a binary or multiclass text classifier on your own data using any of the supported base models (ConfliBERT, BERT, RoBERTa, ModernBERT, DeBERTa, DistilBERT).

Usage

```
conflibert_finetune(
  train,
  dev,
  test,
  model = "ConfliBERT",
  task = "binary",
  epochs = 3,
  batch_size = 8,
  lr = 2e-05,
  save_dir = NULL,
  use_lora = FALSE,
  lora_rank = 8,
  lora_alpha = 16,
  seed = 42
)
```

Arguments

train	A data.frame with text and label columns (training set).
dev	A data.frame with text and label columns (validation set).
test	A data.frame with text and label columns (test set).
model	Base model name. One of: "ConfliBERT", "BERT Base Uncased", "BERT Base Cased", "RoBERTa Base", "ModernBERT Base", "DeBERTa v3 Base", "DistilBERT Base". Default: "ConfliBERT".
task	"binary" or "multiclass". Default: "binary".
epochs	Number of training epochs. Default: 3.
batch_size	Training batch size. Default: 8.
lr	Learning rate. Default: 2e-5.
save_dir	Optional directory to save the trained model. If provided, the model and tokenizer are saved there and can be loaded later with conflibert_load.
use_lora	If TRUE, fine-tune with a LoRA adapter (parameter-efficient; cuts GPU memory roughly 5x). The adapter is merged into the base model before saving, so loading is the same as for full fine-tuning. Default: FALSE.
lora_rank	LoRA rank. Default: 8.

lora_alpha	LoRA alpha. Default: 16.
seed	Random seed for reproducibility. Seeds the classifier-head initialization, data shuffling, and dropout so that two runs with the same seed on the same hardware and package versions give identical results. Change it to study run-to-run variability. Default: 42.

Value

An object of class "conflibert_finetune" (a list, so all existing $ access keeps working) with:

metrics Tibble of test-set metrics.

runtime Training time in seconds.

predictions Integer vector of predicted class labels.

probabilities Matrix of class probabilities (rows = samples, columns = classes).

true_labels Integer vector of true labels.

model_dir Path where the model checkpoint was saved.

model, task The base model name and task type.

It has a themed print() method and a plot() method showing the test-set confusion matrix.

Examples

```
train <- data.frame(
  text  = c("Troops advanced.", "Nice weather today."),
  label = c(1L, 0L)
)
result <- conflibert_finetune(train, dev = train, test = train, epochs = 1)
result$metrics
```

---

conflibert_install *Install Python dependencies for ConfliBERT*

Description

Creates a Python environment and installs the packages needed to run ConfliBERT models (torch, transformers, and friends). Only needs to be run once.

Usage

```
conflibert_install(envname = "conflibert", method = "auto", qa = FALSE)
```

Arguments

envname	Name of the environment. Default: "conflibert".
method	"auto" (default; uses conda when available), "conda", or "virtualenv".
qa	If TRUE, also install TensorFlow. The published ConfliBERT QA checkpoint only ships TensorFlow weights; conflibert_qa converts them to PyTorch on first use (a one-time step that needs TensorFlow) and caches the result, after which TensorFlow is never used again. All other functions are pure PyTorch. Default: FALSE.

Details

As of conflibertR 0.5.0 the backend is PyTorch-only: TensorFlow is no longer required, which makes installation considerably smaller and more reliable. If you have issues with the default virtualenv method, try method = "conda" instead.

Models downloaded from 'Hugging Face' are cached under $HF_HOME (default ~/.cache/huggingface) and the converted QA weights under ~/.cache/conflibertR (or $XDG_CACHE_HOME). Set those environment variables to relocate the caches.

Value

Invisible NULL. Called for its side effect.

Examples

```
## Not run:
# Not run automatically anywhere: this installs software (it creates
# a Python environment and downloads several GB of dependencies), so
# it must only ever be run deliberately by the user.
conflibert_install()

# Include TensorFlow for the one-time QA weight conversion:
conflibert_install(qa = TRUE)

## End(Not run)
```

---

conflibert_load *Load a Fine-tuned Classifier*

Description

Load a classifier saved by conflibert_finetune (when save_dir was specified) or conflibert_active_save. Returns a reusable classifier object you can pass to predict.

Usage

```
conflibert_load(dir)
```

Arguments

dir	Directory containing a HuggingFace checkpoint (config.json, model weights, tokenizer files).

Value

An object of class "conflibert_classifier": a list with model, tokenizer, num_labels, and dir.

Examples

```
# Fine-tune on the bundled example data, saving the model...
data <- conflibert_example("binary")
dir <- file.path(tempdir(), "my_model")
conflibert_finetune(
  train = data$train, dev = data$dev, test = data$test,
  epochs = 1, save_dir = dir
)

# ...then reload it later and predict
clf <- conflibert_load(dir)
predict(clf, c("Troops advanced on the capital.", "Nice weather."))
```

---

conflibert_models *List Available Models*

Description

Returns the names of base models that can be used for fine-tuning and comparison.

Usage

```
conflibert_models()
```

Value

A character vector of model names.

Examples

```
conflibert_models()
```

conflibert_multilabel *Multilabel Classification*

Description

Score text against four event categories: Armed Assault, Bombing or Explosion, Kidnapping, and Other. Each category is scored independently. Inference is batched.

Usage

```
conflibert_multilabel(text)
```

Arguments

text	A character vector of one or more texts.

Value

A tibble (with themed print and plot methods) with columns:

doc_id Integer index of the input text (only when `length(text) > 1`).

text The input text.

label Event category name.

probability Score between 0 and 1.

predicted Logical, TRUE if probability >= 0.5.

Examples

```
conflibert_multilabel("Insurgents kidnapped two aid workers near the border.")
```

conflibert_ner *Named Entity Recognition*

Description

Identify persons, organizations, locations, weapons, and other entity types in text using the pre-trained ConfliBERT NER model. Inference is batched, and printing highlights each entity in its source sentence.

Usage

```
conflibert_ner(text)
```

Arguments

text	A character vector of one or more texts to analyze.

Value

A tibble (with a themed print method) with columns:

doc_id Integer. Which input text this entity came from (only present when length(text) > 1).

entity Character. The entity text.

label Character. Entity type (Person, Organisation, Location, Weapon, etc.).

score Numeric. Mean model confidence for the entity span.

start, end Integer. 1-based inclusive character offsets of the entity in the input text (use with substr()).

Examples

```
conflibert_ner("The soldiers attacked the village near Kabul.")

conflibert_ner(c(
  "NATO forces were deployed to the region.",
  "The UN Security Council met in New York."
))
```

---

conflibert_qa *Question Answering*

Description

Extract an answer from a context passage given a question. Both arguments are vectorized: pass equal-length vectors to answer many questions in one call, or a single context with several questions (scalars are recycled).

Usage

```
conflibert_qa(context, question, details = FALSE)
```

Arguments

context	Character vector of passages.
question	Character vector of questions.
details	If TRUE, return a tibble with the answer plus the model's confidence score and the character span of the answer inside the context. Default FALSE.

Details

The published question-answering checkpoint ships only 'TensorFlow' weights. The first call converts them to 'PyTorch' once and caches the result under `~/.cache/conflibertR` (or `$XDG_CACHE_HOME` when set); subsequent calls reuse the cache and never touch 'TensorFlow' again. Downloaded 'Hugging Face' models are cached under `$HF_HOME` (default `~/.cache/huggingface`). Both caches are written only on explicit, network-enabled calls, and can be relocated by setting those environment variables.

Value

With `details = FALSE` (default), a character vector of answers (a single string when one question is asked, exactly as in previous versions). With `details = TRUE`, a tibble with columns `question`, `answer`, `score`, `start`, `end`, and `context`.

Examples

```
conflibert_qa(
  context  = "The ceasefire was signed in Geneva on March 15th.",
  question = "Where was the ceasefire signed?"
)

# Several questions against one passage, with scores:
conflibert_qa(
  context  = "The ceasefire was signed in Geneva on March 15th.",
  question = c("Where was the ceasefire signed?",
               "When was the ceasefire signed?"),
  details  = TRUE
)
```

`conflibert_status` *Check the conflibertR Setup*

Description

Run a quick diagnostic of the Python backend: whether the `"conflibert"` environment exists, which Python is active, which required packages are importable, and what compute device will be used. Prints a checklist and gives specific advice when something is missing.

Usage

```
conflibert_status()
```

Value

Invisibly, a list with `env_found`, `python`, `packages` (named logical vector), `device`, and `ok`.

Examples

```
conflibert_status()
```

`plot.conflibert_al_session`

Plot an Active Learning Session

Description

Visualize how the model is improving across rounds. The default view is a two-panel plot: the learning curve on top (metrics vs training set size) and the uncertainty trend on the bottom (mean uncertainty of each round's queries). The uncertainty trend is a useful signal: when it flattens, the model is no longer finding informative samples.

Usage

```
## S3 method for class 'conflibert_al_session'
plot(x, which = c("all", "metrics", "uncertainty"), ...)
```

Arguments

`x`	A `conflibert_al_session` object.
`which`	One of `"all"` (default, two panels), `"metrics"` (learning curve only), or `"uncertainty"` (uncertainty trend only).
`...`	Passed to the underlying plot call.

Value

The session, invisibly.

`plot.conflibert_classify`

Plot Classification Results

Description

Base-graphics aggregated view of a `conflibert_classify` result: texts per class with average confidence. For a ggplot2 version use `ggplot2::autoplot()`.

Usage

```
## S3 method for class 'conflibert_classify'
plot(x, ...)
```

Arguments

x	A conflibert_classify tibble.
...	Ignored.

Value

The object, invisibly.

---

plot.conflibert_comparison

Plot a Model Comparison

Description

Base-graphics dot chart of metrics per model for a conflibert_compare result. For a ggplot2 version use ggplot2::autoplot().

Usage

```
## S3 method for class 'conflibert_comparison'
plot(x, ...)
```

Arguments

x	A conflibert_comparison tibble.
...	Ignored.

Value

The object, invisibly.

---

plot.conflibert_finetune

Plot a Fine-tuning Confusion Matrix

Description

Base-graphics test-set confusion matrix for a conflibert_finetune result. For a ggplot2 version use ggplot2::autoplot().

Usage

```
## S3 method for class 'conflibert_finetune'
plot(x, ...)
```

Arguments

x	A conflibert_finetune result.
...	Ignored.

Value

The object, invisibly.

---

plot.conflibert_multilabel *Plot Multilabel Results*

Description

Base-graphics version of the multilabel plot. With several texts it aggregates (share of texts flagged per category); with a single text it shows that text's category probabilities. For a ggplot2 version use ggplot2::autoplot().

Usage

```
## S3 method for class 'conflibert_multilabel'
plot(x, ...)
```

Arguments

x	A conflibert_multilabel tibble.
...	Ignored.

Value

The object, invisibly.

---

plot.conflibert_ner *Plot NER Results*

Description

Base-graphics aggregated view of a conflibert_ner result: entity counts per type. For a ggplot2 version (including a top-entities view) use ggplot2::autoplot().

Usage

```
## S3 method for class 'conflibert_ner'
plot(x, ...)
```

Arguments

x	A conflibert_ner tibble.
...	Ignored.

Value

The object, invisibly.

`predict.conflibert_classifier` *Predict with a Loaded Classifier*

Description

Run batched inference with a classifier loaded via conflibert_load. Returns a tibble with predicted class, confidence, and one prob_* column per class.

Usage

```
## S3 method for class 'conflibert_classifier'
predict(object, text, batch_size = 32, max_seq_len = 512, ...)
```

Arguments

object	A conflibert_classifier.
text	Character vector of texts to classify.
batch_size	Inference batch size. Default: 32.
max_seq_len	Max token sequence length. Default: 512.
...	Ignored.

Value

A tibble with text, class, confidence, and prob_0..prob_{K-1} columns.

Examples

```
data <- conflibert_example("binary")
dir <- file.path(tempdir(), "my_model")
conflibert_finetune(
  train = data$train, dev = data$dev, test = data$test,
  epochs = 1, save_dir = dir
)
clf <- conflibert_load(dir)
predict(clf, c("Bomb exploded.", "Stock market rose."))
```

`theme_conflibert` *ConfliBERT ggplot2 Theme*

Description

A modern, flat ggplot2 theme used by all of the package's `autoplot()` methods: no tick marks, no panel box, hairline grid lines only along the data axis, bold left-aligned titles, and generous whitespace. Use it on your own plots to match.

Usage

```
theme_conflibert(
  base_size = 12,
  base_family = "",
  grid = c("xy", "x", "y", "none")
)
```

Arguments

`base_size`	Base font size. Default: 12.
`base_family`	Base font family. Default: system sans.
`grid`	Which grid lines to keep: `"xy"` (default), `"x"` (vertical only, good for horizontal bars), `"y"` (horizontal only, good for line/column charts), or `"none"`.

Value

A ggplot2 theme object.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  ggplot2::ggplot(mtcars, ggplot2::aes(wt, mpg)) +
    ggplot2::geom_point(colour = "#0ea5e9", size = 3) +
    theme_conflibert(grid = "y")
}
```

Index