# Package ‘safetensors’

August 18, 2025

**Title** Safetensors File Format

**Version** 0.2.0

**Description** A file format for storing tensors that is secure (doesn't allow for code execution), fast and simple to implement. 'safetensors' also enables cross language and cross frameworks compatibility making it an ideal format for storing machine learning model weights.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Suggests** testthat (>= 3.0.0), torch (>= 0.11.0)

**Config/testthat/edition** 3

**Imports** cli, jsonlite, R6, rlang

**URL** `https://github.com/mlverse/safetensors`,
`https://mlverse.github.io/safetensors/`

**BugReports** `https://github.com/mlverse/safetensors/issues`

**NeedsCompilation** no

**Author** Daniel Falbel [aut, cre],
Sebastian Fischer [ctb],
Posit [cph]

**Maintainer** Daniel Falbel `<daniel@posit.co>`

**Repository** CRAN

**Date/Publication** 2025-08-18 12:20:02 UTC

## Contents

safetensors *Low level control over safetensors files*

## Description

Low level control over safetensors files

Low level control over safetensors files

## Details

Allows opening a connection to a safetensors file and query the tensor names, metadata, etc. Opening a connection only reads the file metadata into memory. This allows for more fined grained control over reading.

## Public fields

`con` the connection object with the file

`metadata` an R list containing the metadata header in the file

`framework` the framework used to return the tensors

`args` additional arguments for tensor creation

`max_offset` the largest offset boundary that was visited. Mainly used in torch to find the end of the safetensors file.

## Methods

### Public methods:

- `safetensors$new()`
- `safetensors$keys()`
- `safetensors$get_tensor()`
- `safetensors$clone()`

**Method** `new()`**:** Opens the connection with the file

*Usage:*

```
safetensors$new(path, ..., framework)
```

*Arguments:*

`path` Path to the file to load

`...` (any)
Additional, framework dependent, arguments to pass to use when creating the tensor. For torch, this is the device, for pjrt the client.

`framework` Framework to load the data into. Currently supports "torch" and "pjrt"

**Method** `keys()`**:** Get the keys (tensor names) in the file

*Usage:*

```
safetensors$keys()
```

**Method** `get_tensor()`**:** Get a tensor from its name

*Usage:*

```
safetensors$get_tensor(name)
```

*Arguments:*

`name` Name of the tensor to load

**Method** `clone()`**:** The objects of this class are cloneable with this method.

*Usage:*

```
safetensors$clone(deep = FALSE)
```

*Arguments:*

`deep` Whether to make a deep clone.

## Examples

```
if (rlang::is_installed("torch") && torch::torch_is_installed()) {
tensors <- list(x = torch::torch_randn(10, 10))
temp <- tempfile()
safe_save_file(tensors, temp)
f <- safetensors$new(temp, framework = "torch")
f$get_tensor("x")
}
```

---

# safetensors_frameworks

*Reflection of supported frameworks*

---

## Description

A reflection of supported frameworks.

## Usage

```
safetensors_frameworks
```

## Format

An object of class `environment` of length 1.

safe_load_file *Safe load a safetensors file*

## Description

Loads an safetensors file from disk.

## Usage

```
safe_load_file(path, ..., framework)
```

## Arguments

| | |
|---|---|
| path | Path to the file to load |
| ... | Additional framework dependent arguments to pass to the tensor creation function. |
| framework | Framework to load the data into. Currently supports "torch" and "pjrt" |

## Value

A list with tensors in the file. The metadata attribute can be used to find metadata the metadata header in the file.

## See Also

safetensors, safe_save_file()

## Examples

```
if (rlang::is_installed("torch") && torch::torch_is_installed()) {
  tensors <- list(x = torch::torch_randn(10, 10))
  temp <- tempfile()
  safe_save_file(tensors, temp)
  safe_load_file(temp, framework = "torch")
}
```

## safe_save_file *Writes a list of tensors to the safetensors format*

### Description

Writes a list of tensors to the safetensors format

### Usage

```
safe_save_file(tensors, path, ..., metadata = NULL)

safe_serialize(tensors, ..., metadata = NULL)
```

### Arguments

| | |
|---|---|
| `tensors` | A named list of tensors. Currently only torch tensors are supported. |
| `path` | The path to save the tensors to. It can also be a binary connection, as eg, created with `file()`. |
| `...` | Currently unused. |
| `metadata` | An optional string that is added to the file header. Possibly adding additional description to the weights. |

### Value

The path invisibly or a raw vector.

### Functions

- `safe_serialize()`: Serializes the tensors and returns a raw vector.

### Examples

```
if (rlang::is_installed("torch") && torch::torch_is_installed()) {
  tensors <- list(x = torch::torch_randn(10, 10))
  temp <- tempfile()
  safe_save_file(tensors, temp)
  safe_load_file(temp, framework = "torch")

  ser <- safe_serialize(tensors)
}
```

## `safe_tensor_buffer` *Get raw buffer from a tensor*

### Description

Convert a tensor object to a raw buffer in the formated expected by safetensors.

### Usage

```
safe_tensor_buffer(x)
```

### Arguments

| | |
|---|---|
| `x` | (any)<br>Tensor object. |

### Value

`(raw)`

## `safe_tensor_meta` *Get metadata from a tensor*

### Description

Get the metadata from a tensor.

### Usage

```
safe_tensor_meta(x)
```

### Arguments

| | |
|---|---|
| `x` | (any)<br>Tensor object. |

### Value

`(list)`

# Index