

# Package ‘onnx’

October 14, 2022

**Type** Package

**Title** R Interface to 'ONNX'

**Version** 0.0.3

**Description** R Interface to 'ONNX' - Open Neural Network Exchange <<https://onnx.ai/>>. 'ONNX' provides an open source format for machine learning models. It defines an extensible computation graph model, as well as definitions of built-in operators and standard data types.

**License** MIT License + file LICENSE

**URL** <https://github.com/onnx/onnx-r>

**BugReports** <https://github.com/onnx/onnx-r/issues>

**Encoding** UTF-8

**Depends** R (>= 3.1)

**Imports** reticulate (>= 1.4)

**RoxygenNote** 7.0.2

**Suggests** testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Yuan Tang [aut, cre] (<<https://orcid.org/0000-0001-5243-233X>>),  
ONNX Authors [aut, cph],  
Facebook, Inc. [cph],  
Microsoft Corporation [cph]

**Maintainer** Yuan Tang <[terrytangyuan@gmail.com](mailto:terrytangyuan@gmail.com)>

**Repository** CRAN

**Date/Publication** 2021-04-16 18:50:02 UTC

## R topics documented:

check . . . . .	2
load_from_file . . . . .	3

load_from_string . . . . .	3
make_attribute . . . . .	4
onnx . . . . .	5
print_readable . . . . .	6

<b>Index</b>	<b>7</b>
--------------	----------

---

check	<i>Check Whether The Proto is Valid</i>
-------	---

---

## Description

This method checks whether a protobuf in a particular type is valid.

## Usage

```
check(proto, ir_version)

## S3 method for class 'onnx_pb2.ModelProto'
check(proto, ir_version = 3L)

## S3 method for class 'onnx_pb2.GraphProto'
check(proto, ir_version = 3L)

## S3 method for class 'onnx_pb2.TensorProto'
check(proto, ir_version = 3L)

## S3 method for class 'onnx_pb2.AttributeProto'
check(proto, ir_version = 3L)

## S3 method for class 'onnx_pb2.NodeProto'
check(proto, ir_version = 3L)
```

## Arguments

proto	The proto
ir_version	The version of the proto

## Examples

```
## Not run:

library(onnx)

# Define a node protobuf and check whether it's valid
node_def <- make_node("Relu", list("X"), list("Y"))
check(node_def)

## End(Not run)
```

---

load_from_file	<i>Loads a binary protobuf that stores onnx model</i>
----------------	---

---

**Description**

Loads a binary protobuf that stores onnx model

**Usage**

load\_from\_file(obj)

**Arguments**

obj                    a file-like object (has "read" function) or a string containing a file name

**Value**

ONNX ModelProto object

---

load_from_string	<i>Loads a binary string that stores onnx model</i>
------------------	---

---

**Description**

Loads a binary string that stores onnx model

**Usage**

load\_from\_string(s)

**Arguments**

s                      string object containing protobuf

**Value**

ONNX ModelProto object

---

make\_attribute            *Make Different Types of Protos*

---

### Description

This includes AttributeProto, GraphProto, NodeProto, TensorProto, TensorValueInfoProto, etc.

### Usage

```
make_attribute(key, value, doc_string = NULL)
```

```
make_graph(nodes, name, inputs, outputs, initializer = NULL, doc_string = NULL)
```

```
make_node(op_type, inputs, outputs, name = NULL, doc_string = NULL)
```

```
make_tensor(name, data_type, dims, vals, raw = FALSE)
```

```
make_tensor_value_info(name, elem_type, shape, doc_string = "")
```

### Arguments

key	The key
value	The value
doc_string	The doc_string
nodes	The nodes
name	The name
inputs	The inputs
outputs	The outputs
initializer	The initializer
op_type	The op type
data_type	The data type
dims	The dimensions
vals	The values
raw	If this is FALSE``, this function will choose the corresponding proto field to store the values. If raw is TRUE, use "raw_data" proto field to store the values, and values should be of type bytes in this case.
elem_type	The element type, e.g. onnx\$TensorProto\$FLOAT
shape	The shape

## Examples

```
## Not run:

library(onnx)

# Define a node protobuf and check whether it's valid
node_def <- make_node("Relu", list("X"), list("Y"))
check(node_def)

# Define an attribute protobuf and check whether it's valid
attr_def <- make_attribute("this_is_an_int", 123L)
check(attr_def)

# Define a graph protobuf and check whether it's valid
graph_def <- make_graph(
  nodes = list(
    make_node("FC", list("X", "W1", "B1"), list("H1")),
    make_node("Relu", list("H1"), list("R1")),
    make_node("FC", list("R1", "W2", "B2"), list("Y"))
  ),
  name = "MLP",
  inputs = list(
    make_tensor_value_info('X' , onnx$TensorProto$FLOAT, list(1L)),
    make_tensor_value_info('W1', onnx$TensorProto$FLOAT, list(1L)),
    make_tensor_value_info('B1', onnx$TensorProto$FLOAT, list(1L)),
    make_tensor_value_info('W2', onnx$TensorProto$FLOAT, list(1L)),
    make_tensor_value_info('B2', onnx$TensorProto$FLOAT, list(1L))
  ),
  outputs = list(
    make_tensor_value_info('Y', onnx$TensorProto$FLOAT, list(1L))
  )
)
check(graph_def)

## End(Not run)
```

## Description

R Interface to ONNX

---

`print_readable`*Print the Human-readable Representation of the Proto Object*

---

**Description**

Print the Human-readable Representation of the Proto Object

**Usage**

```
print_readable(x)
```

**Arguments**

x                    The proto object

**Examples**

```
## Not run:
```

```
library(onnx)
```

```
graph_def <- make_graph(  
  nodes = list(  
    make_node("FC", list("X", "W1", "B1"), list("H1")),  
    make_node("Relu", list("H1"), list("R1")),  
    make_node("FC", list("R1", "W2", "B2"), list("Y"))  
  ),  
  name = "MLP",  
  inputs = list(  
    make_tensor_value_info('X' , onnx$TensorProto$FLOAT, list(1L)),  
    make_tensor_value_info('W1' , onnx$TensorProto$FLOAT, list(1L)),  
    make_tensor_value_info('B1' , onnx$TensorProto$FLOAT, list(1L)),  
    make_tensor_value_info('W2' , onnx$TensorProto$FLOAT, list(1L)),  
    make_tensor_value_info('B2' , onnx$TensorProto$FLOAT, list(1L))  
  ),  
  outputs = list(  
    make_tensor_value_info('Y' , onnx$TensorProto$FLOAT, list(1L))  
  )  
)  
print_readable(graph_def)
```

```
## End(Not run)
```

# Index

check, [2](#)

load\_from\_file, [3](#)

load\_from\_string, [3](#)

make\_attribute, [4](#)

make\_graph (make\_attribute), [4](#)

make\_node (make\_attribute), [4](#)

make\_tensor (make\_attribute), [4](#)

make\_tensor\_value\_info  
    (make\_attribute), [4](#)

onnx, [5](#)

print\_readable, [6](#)