

---

**ltr++**

***Release 2021***

## **Marcos Pontes and The Lemur Project**

**Oct 29, 2021**




**CONTENTS:**

<b>1</b>	<b>Integration</b>	<b>3</b>
1.1	Prerequisites . . . . .	3
1.2	1. C++ . . . . .	3
1.3	2. Python . . . . .	4
1.4	3. Installing . . . . .	5
1.5	4. Building . . . . .	5
<b>2</b>	<b>License</b>	<b>7</b>
<b>3</b>	<b>Need Help?</b>	<b>9</b>



Learning to Rank (LTR) is one of the methods that can be effectively applied to solve the task of creating a ranking model in Information Retrieval. It helps solving IR problems such as document retrieval, collaborative filtering, sentiment analysis, computational advertising etc. LTR method aims at learning a model that given a query and a set of candidate documents finds the appropriate ranking of documents according to their relevancy. LTR method aims at learning a model that given a query and a set of candidate documents finds the appropriate ranking of documents according to their relevancy. **LTR++** is a Learning to Rank open-source library based on [RankLib](#) project. Currently two popular algorithms have been implemented and it also implements many retrieval metrics.



img/ltr\_flow.png

The LTR++ project is in early steps of development. Feel free to contribute or suggest features!



## INTEGRATION

### 1.1 Prerequisites

The main prerequisite for installing ltr++ is CMake 3.15 or higher. Other external libraries will be downloaded automatically, if you don't have them. So that make sure you have CMake installed. Also, make sure that your operating system is compatible with any build that is working:

System	1.0.0
Windows x86	
Windows x64	
MacOSX x64	
Linux x64	

### 1.2 1. C++

Using CMake and certifying that all prerequisites are ok, let's understand how to build and install ltr.

#### 1.2.1 Including header-only

After installing the ltr library, you can import the library as a header-only file:

```
1 #include <ltr.hpp>
```

**Warning:** Make sure you have C++17 or higher installed.

#### 1.2.2 Embed as CMake subdirectory

You can use ltr++ directly in CMake projects as a subproject.

Clone the whole project inside your own project:

```
1 $ git clone https://github.com/marcosfpr/ltrpp/
```

and add the subdirectory to your CMake script:

```
1 add_subdirectory(ltrpp)
```

When creating your executable, link the library to the targets you want:

```
1 add_executable(my_target main.cpp)
2 target_link_libraries(my_target PRIVATE ltr)
```

Your target will be able to see the ltr++ headers now.

### 1.2.3 Embed with CPM.cmake

**CPM.cmake** is a nice wrapper around the CMake `FetchContent` function. Install **CPM.cmake** and then use this command to add ltr++ to your build script:

```
1 CPMAddPackage(
2     NAME ltr
3     GITHUB_REPOSITORY marcosfpr/ltrpp
4     GIT_TAG origin/master # or whatever tag you want
5 )
6 # ...
7 target_link_libraries(my_target PUBLIC ltr)
```

Your target will be able to see the ltr++ headers now.

### 1.2.4 Find as CMake package

If you are using CMake and have the library installed on your system, you can then find ltr++ with the usual `find_package` command:

```
1 find_package(ltr REQUIRED)
2 # ...
3 target_link_libraries(my_target PUBLIC ltr)
```

Your target will be able to see the ltr++ headers now.

**Warning:** There is no easy default directory for `find_package` on windows. You have to [set it](#) yourself.

## 1.3 2. Python

Development stage



## 1.4 3. Installing

Development stage

## 1.5 4. Building

### 1.5.1 4.1 Update C++

Update your C++ compiler to at least C++17:

**Ubuntu:**

```

1 # install GCC10
2 sudo apt install build-essential
3 sudo add-apt-repository ppa:ubuntu-toolchain-r/test
4 sudo apt-get update
5 sudo apt install gcc-10
6 sudo apt install g++-10
7 sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 10
8 sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-10 10
9 # Choose gcc-10 there as the default compiler
10 update-alternatives --config g++

```

**Mac OS:**

```

1 # Download clang
2 curl --output clang.tar.xz -L https://github.com/llvm/llvm-project/releases/download/
↳ llvmorg-11.0.0/clang+llvm-11.0.0-x86_64-apple-darwin.tar.xz
3 mkdir clang
4 tar -xvJf clang.tar.xz -C clang
5 # Copy the files to use/local
6 cd clang/clang+llvm-11.0.0-x86_64-apple-darwin
7 sudo cp -R * /usr/local/
8 # Make it your default compiler
9 export CXX=/usr/local/bin/clang++

```

**Windows:**

Update your [Visual Studio Compiler](#)

### 1.5.2 4.2 Building the projet

After installing or updating the dependencies, clone the project with:

```

1 git clone https://github.com/marcosfpr/litrpp.git
2 cd pareto

```

And then build with:

**Ubuntu:**

```
1 mkdir build
2 cd build
3 cmake -version
4 cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="-O2"
5 cmake --build . -j 2 --config Release
6 # The next command for installing
7 sudo cmake --install .
8 # The next command for building the packages / installers
9 sudo cpack .
```

**Mac OS:**

```
1 mkdir build
2 cd build
3 cmake -version
4 cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="-O2"
5 cmake --build . -j 2 --config Release
6 # The next command for installing
7 cmake --install .
8 # The next command for building the packages / installers
9 cpack .
```

**Windows:**

```
1 mkdir build
2 cd build
3 cmake -version
4 cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="/O2"
5 cmake --build . -j 2 --config Release
6 # The next command for installing
7 cmake --install .
8 # The next command for building the packages / installers
9 cpack .
```

## 1.5.3 Python

Development stage

**LICENSE**

Copyright (c) 2021 Marcos Pontes

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



---

CHAPTER  
**THREE**

---

**NEED HELP?**

Please contact [marcos.rezende@aluno.ufop.edu.br](mailto:marcos.rezende@aluno.ufop.edu.br) or [mfprezende@gmail.com](mailto:mfprezende@gmail.com)